Post Processor For Design Of Reinforced Concrete Space Frames Using Object Oriented Programming

by

Jayendra R. Patel

Thesis submitted to the Faculty of the

Virginia Polytechnic Institute and State University

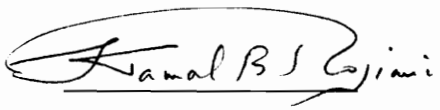
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

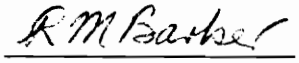
in

Civil Engineering

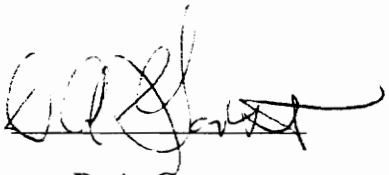
Approved:



K. B. Rojiani



R. M. Barker



D. A. Garst

May, 1994

Blacksburg, Virginia

Post Processor For Design Of Reinforced Concrete Space Frames Using Object Oriented Programming

by

Jayendra R. Patel

Committee Chairman: Dr. Kamal B. Rojiani

Charles E. Via, Jr. Department of Civil Engineering

Virginia Polytechnic Institute and State University

Abstract

A Windows based post processor for the design of reinforced concrete space frames is developed. The post processor is capable of designing beams and columns of reinforced concrete space frames in accordance with the ACI specification. The program is developed in C++ using the object oriented programming approach. The objects used in the program represents a one to one analogy with objects in the real world. The computer model of a structure is composed of objects like members, joints, loads, beams and columns. The development of the post processor is discussed and the program architecture is presented. The design results obtained from the post processor are compared with those obtained from several commercial structural design programs to ensure the correctness of the design. It was concluded that the application of object oriented programming techniques results in programs that are easier to develop and maintain and also greatly reduces the effort required for developing applications for graphical user interfaces.

Acknowledgments

I wish to express my sincere gratitude to my advisor Dr. Kamal Rojiani for his continuous support, patience and guidance throughout this study. I would like to thank Dr. R. M. Barker and Prof. D. A. Garst for serving as committee members. I would also like to thank Tariq Fayyaz for his valuable suggestions for improving the post processor.

Finally, I would like to express my deep gratitude to my father for his love and encouragement. I would also like to thank my brother for his financial and moral support.

Table of Contents

List of Figures

List of Tables

Chapter 1
Introduction

1.1 Introduction

Computer programs for structural engineering applications are typically quite complex and require a large amount of programming effort. In the past, there was greater emphasis on developing codes for structural engineering applications that were efficient in their use of computer memory and hardware resources. This was motivated by the fact that computers had significant memory and hardware limitations and processing time was expensive. With the dramatic improvements in computer hardware, there is now a change in emphasis from developing memory efficient programs to developing programs that are easier to read and to maintain. This change is a direct result of the fact that programmer time has became more expensive than hardware costs. Today, the programmer time required for developing any software constitutes a major portion of the total cost of software. Another factor that contributes to the high cost of software development is the requirement that the software run in a user friendly graphical windowing environment. Software development for graphical user interface (GUI) is considerably more difficult than for character-based environments.

In recent years, many software developers have started using object oriented programming techniques. One of the reasons for this interest is that the characteristics of the object oriented programming paradigm such as, encapsulation of data, abstraction, and inheritance, make it possible to develop programs that are reusable and easier to maintain. Another important advantage of using object oriented programming is that most object oriented programming languages provide a rich collection of class libraries that greatly

reduce the effort required for developing applications for graphical user interfaces such as Microsoft Windows [Kulkarni and Rojiani, 1994].

C++ is the most popular object oriented programming language. Being a superset of C, C++ supports both procedural as well as object oriented programming. This has made the transition from procedural programming to object oriented programming easier for C programmers and is one of the reasons for the growing popularity of C++. Most of the popular C++ compilers come with class libraries for performing many of the routine programming tasks and for developing applications for graphical user interfaces. These class libraries make it considerably easier for programmers to develop applications. They also result in more reliable and robust programs since these class libraries have undergone extensive testing.

Considering the many benefits that both object oriented programming and graphical user interfaces have to offer, it is important to investigate the application of object oriented programming techniques for developing structural engineering applications in a graphical windowing environment. The C++ programming language was selected as the programming language for this research because of its popularity and universal acceptance. The Windows graphical user interface was selected for the same reasons.

1.2 Purpose

The objective of the thesis was to develop a Windows based post processor for the design of reinforced concrete space frames. The post processor is capable of designing the beams and columns in plane and space frames in accordance with ACI specifications [American Concrete Institute, 1990]. In order to provide an integrated application for both analysis and design, the post processor is capable of reading the analysis results from a structural analysis program and requires a minimum amount of input beyond that already

provided during analysis and modeling of the structure. During the development of the post processor, the usefulness of using object oriented programming techniques was also studied.

1.3 Organization

An overview of object oriented programming and its application in structural engineering is presented in Chapter 2. Chapter 3 describes the post processor developed for the design of reinforced concrete space frames. This chapter also serves as a users manual for the program. Chapter 4 focuses on the program architecture with a brief description of the classes used in the program. A comparison of results obtained using this program with those from several commercial programs is presented in Chapter 5. A summary of the research and its major conclusions are given in Chapter 6. Complete definitions of the classes used in this program are provided in Appendix A. The input and output files for two test problems are provided in Appendix B.

Chapter 2

An Overview of Object Oriented Programming

2.1 Introduction

In this chapter an overview of object oriented programming is presented. Different programming paradigms are briefly discussed. Two key concepts: classes and objects, are discussed followed by a description of the concepts of inheritance, polymorphism and reusability. The advantages and shortcomings of object oriented programming are also discussed. A review of the literature on the application of object oriented programming in structural engineering is also presented.

2.2 Different programming paradigms

A programming paradigm is the way of solving a programming problem. Some examples of programming paradigms are procedural programming, rule based programming, serial programming, parallel programming and object oriented programming. Each of these programming paradigms is based upon its own conceptual framework. In the procedural programming paradigm, the problem is divided into functions with each function performing a clearly defined task. Several functions that carry out a specific job are grouped together into modules. In rule-based programming, a series of rules are defined that operate on the data. In serial programming paradigm, only one function is executed at any particular time, while in parallel programming paradigm, concurrent functions are executed simultaneously. In the object oriented programming paradigm, both data and functions that operate on that data, are combined together into a single unit called an object. Each of the above programming paradigms has its advantages

and disadvantages and there is no single programming paradigm that is best for all types of applications. For example, the rule-based programming paradigm would be well suited for the design of a knowledge base while object oriented programming paradigm is better suited for applications where complexity is the main issue.

2.3 Object Oriented Programming Paradigm

Object oriented programming is relatively new. In object oriented programming, the problem is divided into objects that correspond to the real world. "Object oriented programming can be considered as the viewing of a program as an analog model of a real-world phenomenon. A program thus builds an environment containing objects (such as beams, floors and columns) with attributes analogous to the real-world counterparts (such as weight, length and material properties) and analogous behaviors (deformation under load, adding self weight to a structure). Furthermore, objects communicate with each other and the user, and can be given necessary skills and 'intelligence' for carrying out requested tasks" [Miller, 1991].

2.4 Two Key Concepts: Classes and Objects

In object oriented programming, the basic building blocks are classes and objects. The class specifies the data structure and functions that apply to each object of the class. An object is an instance of a class. Many objects of the same class can be created. Thus, the class acts as a template for creating objects of that class. Defining a class does not create any object and hence does not allocate any memory for its data members. Memory is allocated for data members of that object only when an object is created from a class. The objects which exist in memory, embody the characteristics of their classes.

Class members can be declared as private, protected or public. Private members are accessible only by the member functions declared within the same class. Protected members can be accessed by member functions within the same class, and by member functions of classes that are derived from this class. Public members are accessible from outside the object of the class. Typically all data is made private so that it can be accessed only through member functions of that object. The interaction between objects occurs by means of public member functions which are accessible from other objects. Other objects can receive a copy of the private data of another object by sending a request to that object but cannot alter the private data of the object. Thus, in object oriented programming, data is safe from accidental corruption [Lafore, 1991]. In other words, data is hidden inside the object and can only be accessed through functions of that object. Objects interact with each other through a clearly defined interface only. This feature of object oriented programming is called *abstraction.* Combining a data structure with functions dedicated to manipulating the data is called *encapsulation.* Subclasses can be derived from the base class. Each subclass shares characteristics that are common to the base class and also has its own particular characteristics. This is called *inheritance*.

An object performs its task by calling appropriate functions to operate upon its own data. For example, a program for designing beams and columns may contain a beam and a column object. Whenever the program has to design a beam, it sends a message to the beam object. The beam object has all the data and functions needed to perform the design. In the same way, if a column is to be designed, a message is sent to the column object to perform the design. Thus objects are active entities and are self sufficient.

2.5 Inheritance, Polymorphism and Reusability

A new class can be derived from an existing class. In C++, the original class is called the base class and the newly derived class is called the derived class. The base class contains characteristics common to a group of derived classes. The derived class inherits the characteristics of the base class but adds new ones of its own. The base class is not affected by this process. This unique feature is called *inheritance*. Inheritance is of prime importance in object oriented programming. In some object oriented programming languages, derived class can be created from more than one base class. This is called *multiple inheritance*. Inheritance clarifies the relationship among classes, reduces the size of the program and helps in the conceptualization of the program. According to Miller, "One of the more powerful features of object oriented programming is the capability for sharing attributes and skills by means of inheritance" [Miller, 1991].

In the derived class, it is possible to redefine the parent functions as necessary without changing their names. The same function call in the program performs different tasks depending on the type of object. "This process of having the same member function exhibit different behavior based on object type is known as *polymorphism*" [Roetzheim, 1992]. "The concept of polymorphism allows the programmer to develop cleaner programs at a higher level of abstraction. Therefore, readability and modifiability are improved" [Lee and Arora, 1991].

One of the most important advantages of inheritance and polymorphism is code *reusability*. An existing class can be used without modification as the base class to create derived classes that can be used for different situations. Code reusability saves time and money and also increases the program reliability. For example, Object Windows Library, an object oriented class library included with the Borland C++ compiler, provides all of the necessary classes for developing applications for the Microsoft Windows graphical

windowing environment. The effort required to develop Windows applications is significantly reduced when these class libraries are used.

2.6 Object Oriented Programming Languages

There are many object oriented programming languages in use today. Examples of object oriented programming languages include C++, Objective C, SmallTalk, ACTOR, Trelis, Object Pascal, Flavors and Eiffel. C++ is the most widely used object oriented programming language. In addition to C++, some of the more popular object oriented programming languages are SmallTalk, Objective C, Object Pascal and Eiffel.

SmallTalk is the oldest object oriented programming language. It was developed during the 1970s by Kay, Goldberg and Ingalls at Xerox [Hughes, 1991]. The first commercially available version of SmallTalk, SmallTalk-80, was released in 1983. Both C++ and Objective C are supersets of the C programming language. They extend the C programming language for object oriented programming. Eiffel is a relatively new object oriented programming language. Eiffel is loosely based on Simula but offers much greater sophistication [Hughes, 1991]. Turbo Pascal from Borland International and Quick Pascal from Microsoft Corporation extend Pascal for object oriented programming.

In a pure object oriented programming language, every element is an object and all processing is achieved by sending messages to objects. SmallTalk and Eiffel are pure object oriented programming languages. SmallTalk is the most widely used pure object oriented programming language. Programming languages developed by extending conventional procedural programming languages are called hybrid programming languages. C++, Objective C, Turbo Pascal and Quick Pascal are examples of hybrid programming languages. Since these languages support both object oriented programming and procedural programming, they are very popular among the programmers who want to

switch from procedural programming to object oriented programming. C++ is the most popular hybrid programming language.

2.7 Advantages and Shortcomings of Object Oriented Programming

Objects and classes are the key elements of object oriented programming. In object oriented programming, data and procedures are combined to form a class. The concept of a class not only supports abstraction, but also supports modularity. It provides a natural way to modularize the program. For example, a "Beam" class may represent a single module of a frame design program. Together, abstraction and modularity make modeling of an object oriented program very easy and hence increase programmer productivity. Encapsulation hides the details of the implementation of a class. This makes even complex classes easier to use. Data hiding also protects the program from unwanted side-effects that can easily occur when access to program data is not localized.

In an object oriented program there is generally a one-to-one correspondence between objects in the real world and objects in the program. This results in more realistic modeling of the problem. According to Powell et al., "Many objects will correspond directly to entities in the engineering system which is being modeled, narrowing the 'semantic gap' between real world entities and their software representations" [Powell, Abdalla and Sause, 1989]. According to Micallef, "One of the advantages of using object oriented programming languages for building large systems is that they facilitate the creation of software components that closely parallel the application domain, an important feature for building understandable systems" [Micallef, 1988].

Inheritance provides several advantages. With inheritance, new applications can be created by using preexisting well-proven components which have been previously tested and debugged. Software companies now provide class libraries for different applications.

The use of these libraries results not only in increased productivity but also in increased program reliability.

When using object oriented programming the need for very careful advance planning cannot be overemphasized. To take maximum advantage of code reusability, classes should be made as generic as possible. This usually requires additional effort on the part of the programmer. Most object oriented programming languages provide large class libraries for performing many of the routine programming tasks. In order to use these class libraries effectively, it is necessary to become familiar with these libraries.

A disadvantage of object oriented programming is slightly higher memory requirement and longer execution time. This is mainly because of message passing and dynamic linking. However, with the dramatic improvements in hardware, execution speed and memory requirements are not significant factors and are of little importance compared to the advantages offered by object oriented programming.

2.8 Application of Object Oriented Programming in Structural Engineering

Miller considered the application of the object oriented approach in developing software for structural engineering applications [Miller, 1991]. He discussed concurrency issues, data base issues and analysis issues for structural engineering. According to Miller, "Object oriented databases, allow for much more expressive power in defining relations and dependencies, and also make it possible to group data and operations together". In another paper, Miller discussed in depth the issues of object oriented concurrent structural analysis [Miller, 1989]. Miller also developed a LISP based object oriented structural model [Miller, 1988].

Powell focused on data base design for computer-integrated structural engineering [Powell and Bhateja, 1988]. He presented a data base design based on a "component-

connection" abstraction model and the object oriented data model. He also addressed the issue of database management for computer-integrated structural engineering software and discussed number of aspects related to object management [Powell and Abdalla, 1989]. An algorithm for automated stiffness modeling of frame structures using object oriented programming paradigm developed by Powell and others is discussed in Powell and An-Nashif [1991].

Kulkarni developed several applications for the design of reinforced concrete members in Windows environment [Kulkarni, 1993]. One of the applications he developed was an application for the analysis and design of reinforced concrete continuous beams using a pure object oriented language, ACTOR, and a hybrid object oriented language, C++. He compared various aspects of these two languages and concluded that C++ was superior to ACTOR. Adeli and Yu used C++ to develop a program to design steel girders [Adeli and Yu, 1991]. They used an object oriented approach in developing this application. Yu and Adeli also discussed the utilization of object oriented concepts in structural design and presented an object oriented model for the design of welded steel plate girders according to the AISC Load and Resistance Factor Design specification [Yu and Adeli, 1991].

Fordge et al. investigated the suitability of the object oriented programming approach for finite element analysis [Fordge, Foschi and Stiemer, 1990]. The analysis program was described as an assembly of classes which organize and control the solution process. They found that "Implementation of an object oriented program requires less time, results in smaller programs, and provides better management of data and procedures than that of an equivalent procedural program". Dubois-Pelerin et al. also used object oriented programming concepts for the development of a finite element analysis program [Dubois-Pelerin, Zimnermann and Bomme, 1992]. The main classes used in the simple

linear static program they developed were: Element, Node, Material, Load and Structure. Baugh and Rehak presented examples of various abstractions, implementation issues and their use in finite element programming [Baugh and Rehak, 1989]. Remy et. al. used an object oriented programming paradigm to develop an interactive user interface for a finite element analysis program [Remy, Devloo and Filho, 1991]. Abdalla and Yoon developed a general purpose data-translation facility for finite element and graphics-based programs using the object oriented approach [Abdalla and Yoon, 1992].

Garrett described an object oriented representation for design standards [Garrett, 1989]. Hakim and Garrett [1992] discussed the usefulness of the object oriented representation in general and civil engineering knowledge representation in particular. They also discussed the use of the object oriented concepts for the next generation of database management systems. Akhras and Foo described the design and development of two object oriented systems in relation to diagnostic problems in timber buildings and pavements [Akhras and Foo, 1993]. Bierdermann and Grierson illustrated the principles and advantages of object oriented programming by developing an application to design floor systems [Bierdermann and Grierson, 1992]. Jaiswal and Riggs developed an object model for structural analysis and design of two dimensional, steel frame structures. This model can handle only static loads and has only limited design and drawing capabilities [Jaiswal and Riggs, 1991].

In this chapter, an overview of object oriented programming was provided. Different programming paradigms were presented. The concepts of classes, objects, inheritance, polymorphism and reusability were briefly described and the various object oriented programming languages were discussed. The advantages and disadvantages of object oriented programming were also discussed. In the last section, a review of the

literature on the application of object oriented programming for structural engineering applications was presented.

Chapter 3
The Post Processor

3.1 Introduction

In this chapter, a description of the post processor for the design of reinforced concrete space frames is presented. The procedure used for the design of beams and columns is described. Also, the default design parameters and assumptions made during the design process are presented. This chapter also serves as a users manual for the program. The various menus and program options and dialog boxes for obtaining user input are presented along with a description of the output available from the post processor.

3.2 Overview of the Design Procedure

The post processor designs reinforced concrete frames in accordance with the ACI 318-89 specifications [American Concrete Institute, 1990]. By allocating memory dynamically, it is possible to handle any number of members, load cases and load combinations. Thus, the size of the frame that can be designed is limited only by the available memory. Default values are provided in most cases in order to minimize data entry. However, these default values can easily be modified. Dialog boxes and menus are used to facilitate the data entry process. The input to the program consists of the type of structure (i.e., plane frame or space frame), number of joints, number of members, number of supports, number of load cases, number of load combinations, joint coordinates, member incidences, member properties (i.e., cross sectional area and moment of inertia), joint releases if any, applied loads and member end forces. It is not necessary to enter the

above information since the program reads it in from the output file created by the CADKEY frame analysis program which already contains all of this information.

3.2.1 Default Beam Design Parameters

Default values for the compressive strength of concrete, yield strength of steel, clear cover and size of the stirrups are used for the beam design. Also, the program uses the section dimensions specified during the analysis from the CADKEY analysis output file. The default design parameters can be modified during the design process. Also, new section dimensions can be selected during the design. Table 3.1 shows the default values provided by the program.

Table 3.1 Default Beam Design Parameters

Parameter	Default Value
Concrete compressive strength, $f_c^{'}$	4.00 ksi
Yield strength of steel, f_y	60.0 ksi
Clear cover to the reinforcement	1.5 in.
Stirrup size (U shaped, vertical)	# 4

3.2.2 Default Column Design Parameters

The program also provides default values for the compressive strength, yield strength of steel and clear cover for column design. Again the section dimensions from the CADKEY analysis output file are used. The column dimensions cannot be changed during the design process. Table 3.2 shows the default values provided by the program. The

program designs the columns with longitudinal reinforcement on all four sides. This can be modified if needed to design columns with reinforcement on two sides only.

Table 3.2 Default Column Design Parameters

Parameter	Default Value
Concrete compressive strength, f_c'	4.00 ksi
Yield strength of steel, f_y	60.0 ksi
Clear cover to the reinforcement	1.5 in.

3.2.3 Mark Groups

A mark group is a group of members that have the same cross-sectional dimensions. Mark groups can be specified for the beams that have similar properties and loading. The program selects the same dimensions for all members in a mark group. Program provides two design options for mark groups; a) same dimensions and b) same dimensions and reinforcement. In the first case, the dimensions of all members are the same, and are based on the member in the mark group that has the largest section dimensions. In the second case, section dimensions and the flexural and shear reinforcement are the same for all members in the mark group. There is no upper limit on the number of beams in a mark group, but there can only be a maximum of ten mark groups.

3.2.4 Beam Design

The program first designs all of the beams individually and then designs the beams in mark groups. The critical loading combination is determined for each beam. Shear

forces and positive and negative bending moments are computed at fifteen equally spaced sections along the span. The critical shear near the end of a beam is the shear force at a distance of d from the face of the support, where d is the effective depth. If the option to select new dimensions is in effect then the program suggests a set of suitable dimensions for the beam based on flexure. The beam width and depth can be selected from the suggested dimensions or any other reasonable dimension can be entered. If the option to use the dimensions from the analysis file is in effect, the program uses the cross sectional area and moment of inertia from the analysis file to compute the section dimensions assuming a rectangular section, and uses these dimensions in design.

The adequacy of the section for flexure and shear is checked. A message is displayed if the beam has to be designed as doubly reinforced section. If the user does not want the beam to be doubly reinforced, the program computes new dimensions for the beam. A message is also displayed if the shear check fails. Again, the program computes new dimensions for the section. If both the shear and flexure checks are satisfied and the dimensions are suitable, the program proceeds to determine the flexural and shear reinforcement. For flexural design, the area of steel required at top and bottom of all fifteen sections is determined. For shear design, the spacing of vertical U-shaped stirrups at these fifteen sections is determined.

For mark groups, the critical shear forces and positive and negative bending moments at all sections for all beams in the mark group are computed first. If it is required to have the same section dimensions for all beams in the mark group then the critical bending moment for all beams in that mark group is determined. The section dimensions for this bending moment are computed. These section dimensions are then used for all beams in the mark group. If it is required to have the same section dimensions and reinforcement then the critical shear force and bending moment at each section for all

beams in mark group are computed and all beams in the mark group are designed for these forces, ensuring same flexural and shear reinforcement at all sections.

3.2.5 Column Design

Each column is designed for all load combinations and the load combination requiring the largest area of longitudinal steel for a column is considered to be the critical load combination for that column. The program calculates effective length factors for bending about both axes. The calculation of these factors involves the solution of Equation (1) for the bending condition and Equation (2) for the sway condition [Salmon and Johnson, 1980].

$$f(p) = 0.25\,\Psi_a\,\Psi_b\,(p)^2 + 0.5\,(\Psi a + \Psi b)\,(1 - p\cot(p)) + 2.0\tan(0.5p)/p - 1 \quad (1)$$

$$f(p) = \Psi_a\,\Psi_b\,(p)^2 + 6.0\,(\Psi a + \Psi b)\,p\cot(p) - 36.0 \quad (2)$$

where:

Ψa = ratio of the sum of the column rotational stiffnesses to the sum of the beam rotational stiffnesses at the first end of the member.

Ψb = ratio of the sum of the column rotational stiffnesses to the sum of the beam rotational stiffnesses at the second end of the member.

k = effective length factor

p = π / k

The above equations are solved using a numerical iterative technique. The program also calculates the unsupported lengths for bending about both axes of the column while calculating the effective length factors.

A check for slenderness is also performed. If the column is found to be a long column, moment magnification factors are computed according to Section 10.11.5 of the ACI code. The resultant uniaxial moment is calculated for biaxially loaded columns. The column is designed for the resultant moment and the adequacy of the design is checked using the reciprocal load method. An iterative scheme is used to determine the required area of longitudinal steel for each load combination.

The sizes and number of bars are found for the largest steel area required. The program determines all possible bar combinations considering all bar sizes ranging from #4 to #18, which give areas of steel within 140 percent of the required steel area. A maximum of two bar sizes is used when selecting bars and only bars that are no more than two bar sizes apart are considered. Also, a maximum of five bars of each size is used when there are two different bar sizes and a maximum of ten bars for the case when there is only one bar size considered. The program checks to make sure that the bars do indeed fit within the section without violating ACI code requirements regarding cover and bar spacing. If a single bar size is found that gives a steel area within 5% of the required steel area, then it is selected first, otherwise the bar combination consisting of two bar sizes that gives the least area of steel is selected.

After the longitudinal steel has been selected, lateral ties are designed according to Section 7.10.5 of the ACI code. The size of the ties and the spacing of the ties is determined.

3.2.6 Design Results

Design results can be displayed on the screen. They can also be saved in a file or can be printed directly on the printer.

3.2.6.1 Beam Design Results

The beam design results consist of the member number, group number in the case of a beam belonging to a mark group, width and depth of the section, compressive strength of concrete f_c', yield strength of steel f_y, stirrup size to be used, design moments and shear forces at fifteen sections and flexural steel area and spacing of stirrups at each section.

3.2.6.2 Column Design Results

The column design results consist of number and size of longitudinal reinforcement and the size and spacing of the ties. Design results also include the member number, section dimensions, compressive strength of concrete f_c', yield strength of steel f_y, the critical load combination for the design, and the applied end moments and axial load for the critical loading combination. Calculated values of effective length factors (k_{bx} and k_{by} for bending and k_{sx} and k_{sy} for sway) and moment magnification factors (δ_{bx} and δ_{by} for bending and δ_{sx} and δ_{sy} for sway) are also shown.

3.3 Program Operation

This post processor runs under the Windows environment. Menus are used to facilitate navigation between the various parts of the program. Dialog boxes are used to obtain user input. Separate windows are used for viewing the structure, beam design results and column design results.

3.3.1 Start Menu

When the application is started, the Start Menu is assigned to the application's main window. The Start Menu consists of two pull down menus, File and Help. The File

menu has two menu items, Open and Exit, and the Help menu has one menu item, About. Fig. 3.1 shows the screen when menu item About of the Help menu is selected.

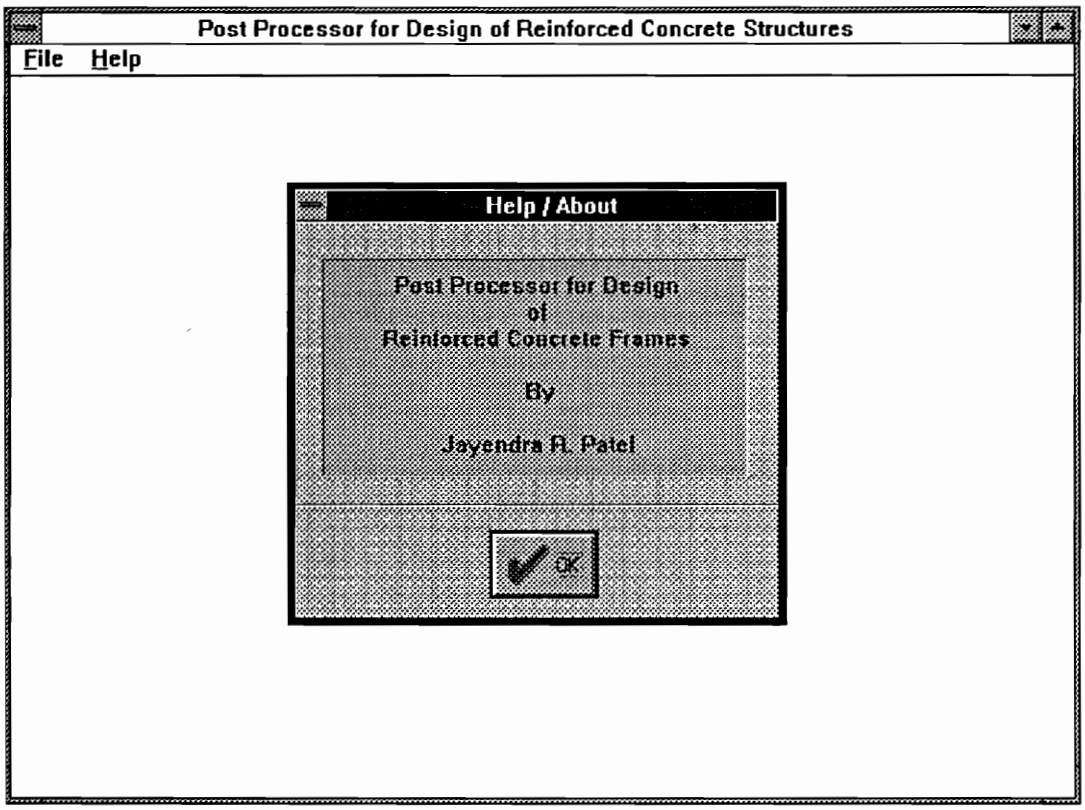


Fig. 3.1 Start Menu

The program terminates when the Exit menu item is selected from the File menu. When the Open menu item of the File menu is selected, the File Open dialog box is activated which allows the user to enter the file name of the input data file. The input file is the output file that is generated by the CADKEY frame analysis program. After reading the input file, the structure is displayed in the main window and the Start Menu is replaced by the Main Menu.

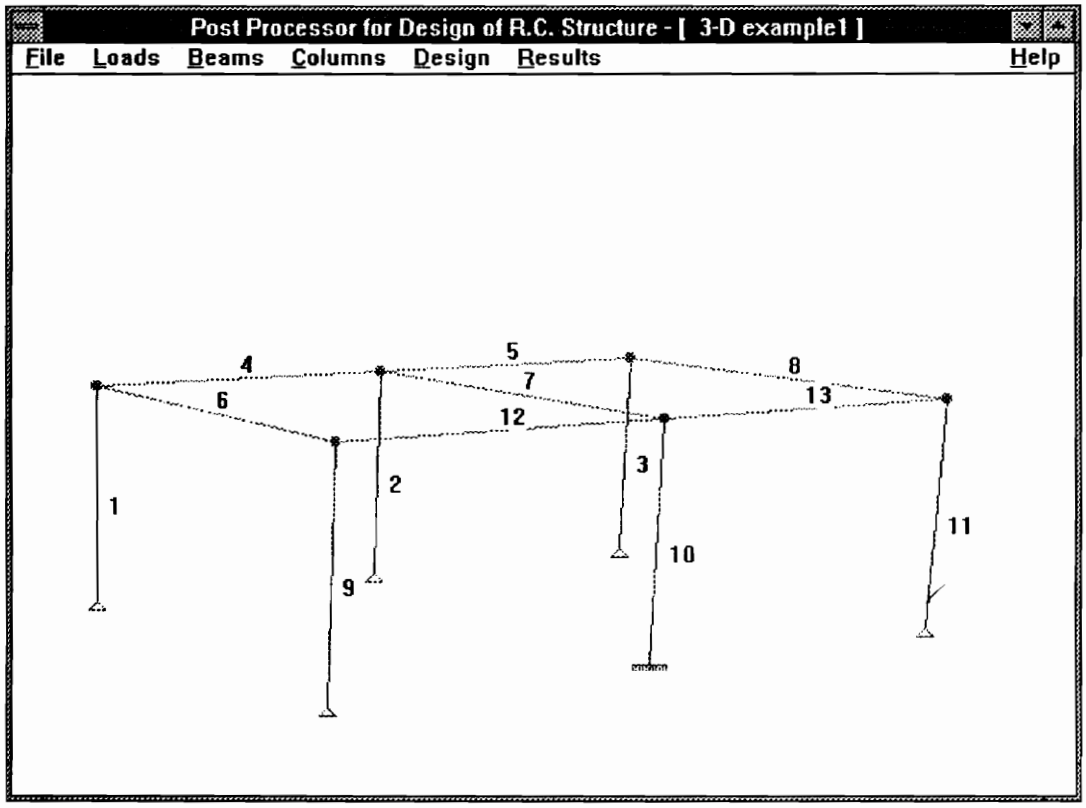


Fig. 3.2 Main Menu

3.3.2 Main Menu

Fig. 3.2 shows the application window with the Main Menu. The Main Menu consists of seven pull down menus: File, Loads, Beams, Columns, Design, Results and Help. These pull down menus allow the user to navigate between different parts of the program and are described below.

3.3.2.1 File Menu

The File menu has five menu items: Open, Save, Save As, Print and Exit. Initially the Open menu item is grayed (disabled). Also, the Menu items, Save, Save As and Print are grayed at the start and become active after beam design or column design has been performed. When the Save menu item is selected, the program creates an output file

having the same file name as the input file but with the extension '.out'. When the Save As menu item is selected, the program displays the File Save As dialog box allowing user to specify the name of the output file. The output file can also be printed by selecting the Print menu item.

3.3.2.2 Loads Menu

The Loads menu has one menu item, Define Load Cases. When selected, the Load Cases dialog box is displayed to allow the user to define the type of load: dead load, live load, wind load, earthquake load or snow load that applies to the current load case. Fig 3.3 shows the Load Cases dialog box.

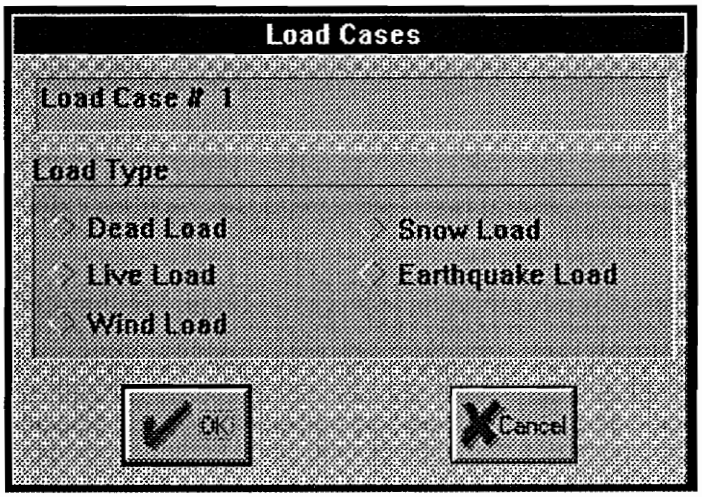


Fig. 3.3 Load Cases Dialog Box

3.3.2.3 Beams Menu

The Beams menu has two menu items: Design Parameters and Mark Groups. The default beam design parameters can be modified by selecting the Design Parameters menu

item. Fig. 3.4 shows the Beam Design Parameters dialog box which is displayed when this menu item is selected. When the OK button of this dialog box is pressed, program checks for the validity of the input data. If invalid input data has been entered, the error message shown in Fig. 3.5 is displayed. If data is valid, the default values are modified and the dialog box is closed. If the Cancel button is pressed, default values are not modified and dialog box is closed.

Mark groups for beams can be assigned by selecting the menu item Mark Groups. Fig. 3.6 shows the Mark Group dialog box.

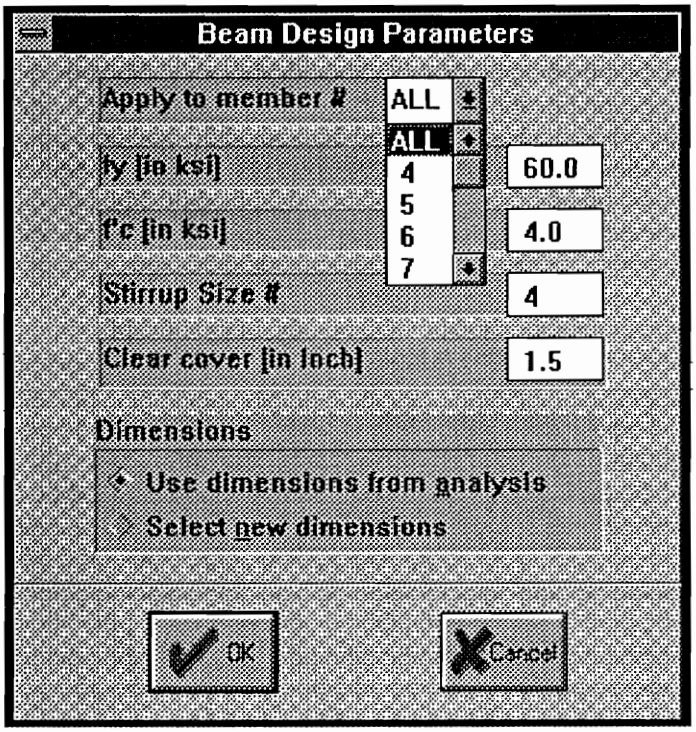


Fig. 3.4 Beam Design Parameters

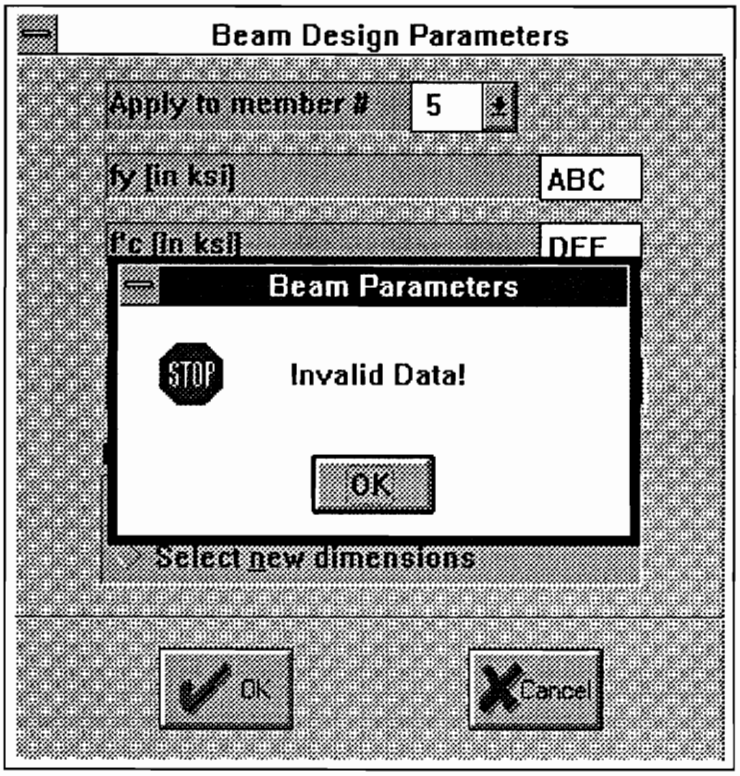


Fig. 3.5 Message Box for Illegal Data

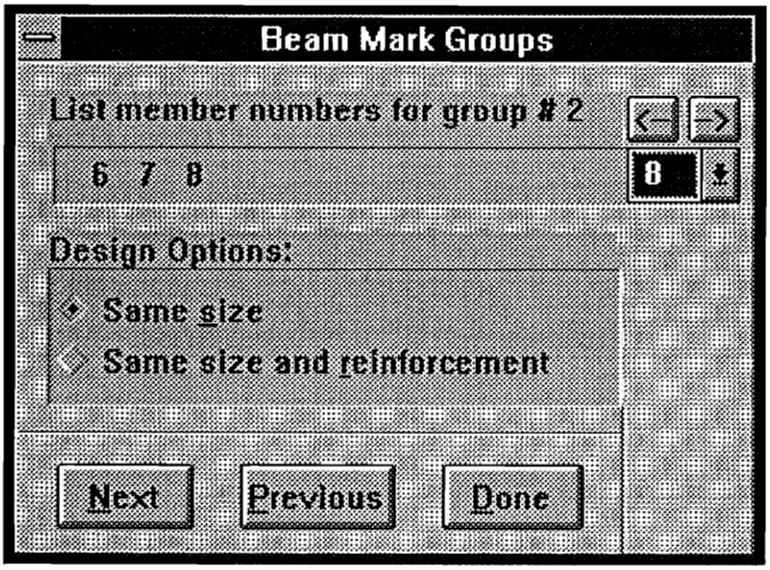


Fig. 3.6 Beam Mark Groups Dialog Box

3.3.2.4 Columns Menu

The Columns menu has one menu item, Design Parameters. By selecting this menu item, the default column design parameters can be modified. Fig. 3.7 shows the Column Design Parameters dialog box.

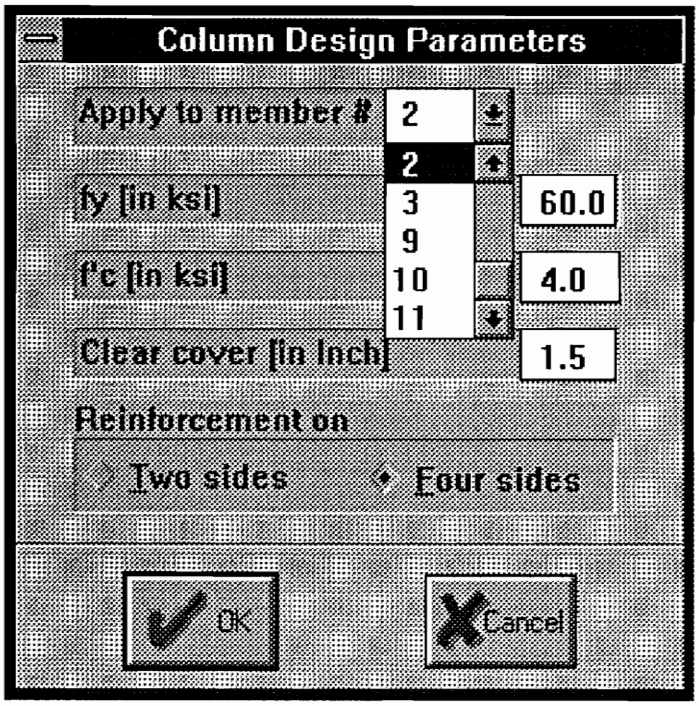


Fig. 3.7 Column Design Parameters

3.3.2.5 Design Menu

The Design menu has two menu items: Beams and Columns. When Beams menu item is selected, all beams in the structure are designed using the design process described in Section 3.2.4. If the option to select new dimensions is active, the program displays a dialog box containing a list of possible member sizes and allows the user to select new dimensions.

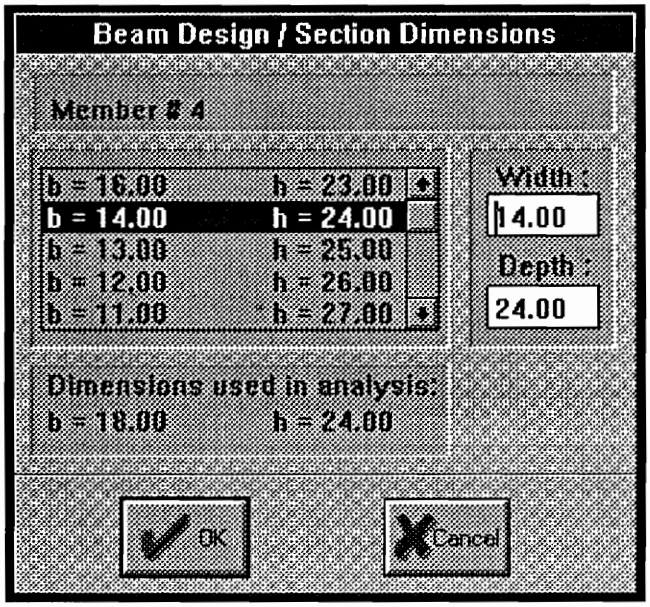


Fig. 3.8 Select Dimensions Dialog Box

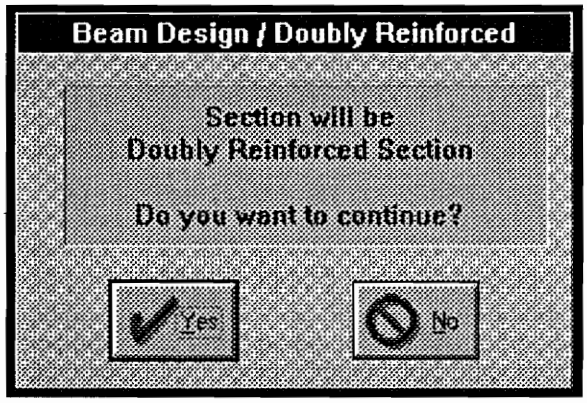


Fig. 3.9 Doubly Reinforced Section Dialog Box

The Select Dimensions dialog box, which is shown in Fig. 3.8 displays a list of suggested section dimensions. The section dimensions can be selected from this list or by entering the width and depth in the appropriate edit fields. The dialog box also shows the

dimensions used in the analysis as a guide for selecting dimensions. If the Cancel button is pressed, the program displays a message box asking whether or not the program should be terminated since the program cannot continue with the design without beam dimensions. If the OK button of this message box is pressed, the program terminates. If the Cancel button of this message box is pressed, the message box disappears and the design process continues. If for selected dimensions, the beam has to be designed as a doubly reinforced beam, the dialog box shown in Fig. 3.9 is displayed, asking user whether or not the section should be designed as a doubly reinforced section. If the Yes button of this dialog box is pressed, the beam is designed as a doubly reinforced section. If the No button of this dialog box is pressed, the user is asked to select new dimensions. If the shear check fails for the selected section dimensions, a message box is displayed requesting the user to select new dimensions. Fig. 3.10 shows this message box.

Fig. 3.10 Shear Check Failed Message Box

Columns in the structure can be designed by selecting the Column menu item of the Design menu. When the Column menu item is selected, the program checks to see whether all load cases have been defined. If a load case has not been defined, the program displays a message box asking the user to define load cases first. Fig. 3.11 shows this message box. Load cases can be defined by selecting the Define Load Cases menu item of

the Loads menu. In order to design a column of an unbraced frame, at least one load case must be of the type dead load. If all of the load cases are defined and there is no load case of the type dead load, the error message shown in Fig. 3.12 is displayed. When all load cases have been defined and there is at least one of type dead load, the program begins the designing process. A message window is displayed on the screen indicating which member is being designed. When all of the columns of the structure have been designed, the message window disappears.

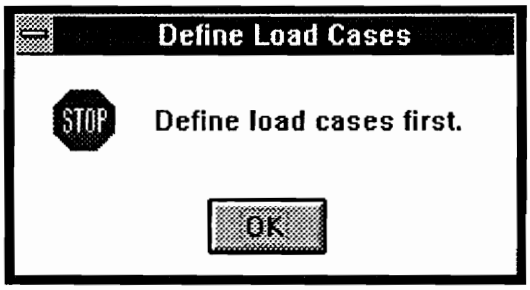


Fig. 3.11 Message Box: Load Cases Not Defined

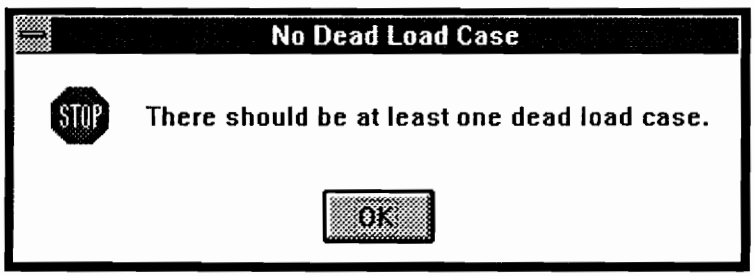


Fig. 3.12 Message Box: No Dead Load

3.3.2.6 Results Menu

The Results menu has two menu items: Beams and Columns. When the Beams menu item is selected, the program displays the Beam Number dialog box allowing the user to specify the member number of the beam for which results are to be viewed. The Beam Number dialog box is shown in Fig. 3.13. The list box in this dialog box contains the member numbers of all beams in the structure. A member number can be selected from the list. After a member number has been chosen from the list, the Beam Design Results child window containing beam design results for the member number chosen is displayed. Fig. 3.14 shows this child window.

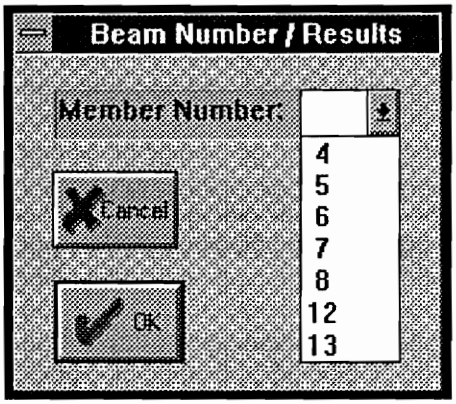
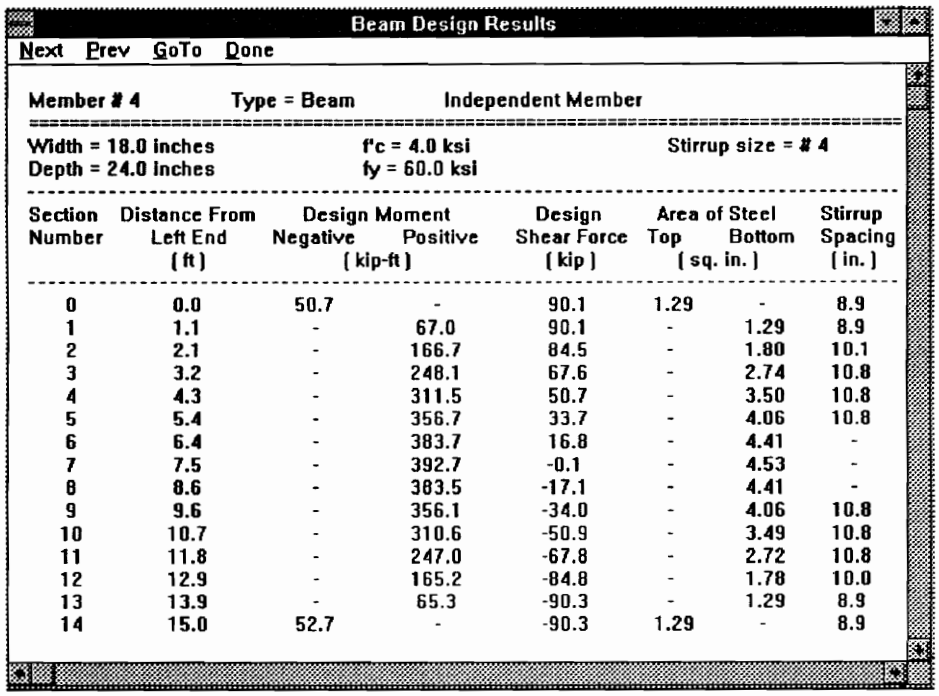


Fig. 3.13 Beam Number Dialog Box

If the Columns menu item is selected, the Column Number dialog box is displayed to allow the user to select the member number of the column for which results are to be viewed. This dialog box is similar to the Beam Number dialog box with the only difference that the list box of the Column Number dialog box contains the member numbers of all columns in the structure. When a member number is chosen from the list box and the OK

button is pressed, the dialog box is closed and the Column Design Results child window containing column design results is displayed as shown in Fig. 3.15.



Beam Design Results

Next Prev GoTo Done

Member # 4 Type = Beam Independent Member

Width = 18.0 inches f'c = 4.0 ksi Stirrup size = # 4
Depth = 24.0 inches fy = 60.0 ksi

Section Number	Distance From Left End (ft)	Design Moment Negative (kip-ft)	Design Moment Positive (kip-ft)	Design Shear Force (kip)	Area of Steel Top (sq. in.)	Area of Steel Bottom (sq. in.)	Stirrup Spacing (in.)
0	0.0	50.7	-	90.1	1.29	-	8.9
1	1.1	-	67.0	90.1	-	1.29	8.9
2	2.1	-	166.7	84.5	-	1.80	10.1
3	3.2	-	248.1	67.6	-	2.74	10.8
4	4.3	-	311.5	50.7	-	3.50	10.8
5	5.4	-	356.7	33.7	-	4.06	10.8
6	6.4	-	383.7	16.8	-	4.41	-
7	7.5	-	392.7	-0.1	-	4.53	-
8	8.6	-	383.5	-17.1	-	4.41	-
9	9.6	-	356.1	-34.0	-	4.06	10.8
10	10.7	-	310.6	-50.9	-	3.49	10.8
11	11.8	-	247.0	-67.8	-	2.72	10.8
12	12.9	-	165.2	-84.8	-	1.78	10.0
13	13.9	-	65.3	-90.3	-	1.29	8.9
14	15.0	52.7	-	-90.3	1.29	-	8.9

Fig. 3.14 Beam Design Results

3.3.3 View Menu

Figures 3.14 and 3.15 show the View Menu. Both, the Beam Design results and the Column Design Results child windows, are assigned the View Menu. The View Menu has four menu items: Next, Prev, GoTo and Done. When the Next menu item is selected, design results for the next beam or column member are displayed. Similarly, if the Prev menu item is selected, the design results for the previous beam or column are displayed. When the GoTo menu item is selected, the Beam Number dialog box or Column Number

dialog box is displayed allowing the user to select the member number for which results are to be viewed. Selection of the Done menu item, closes the results window.

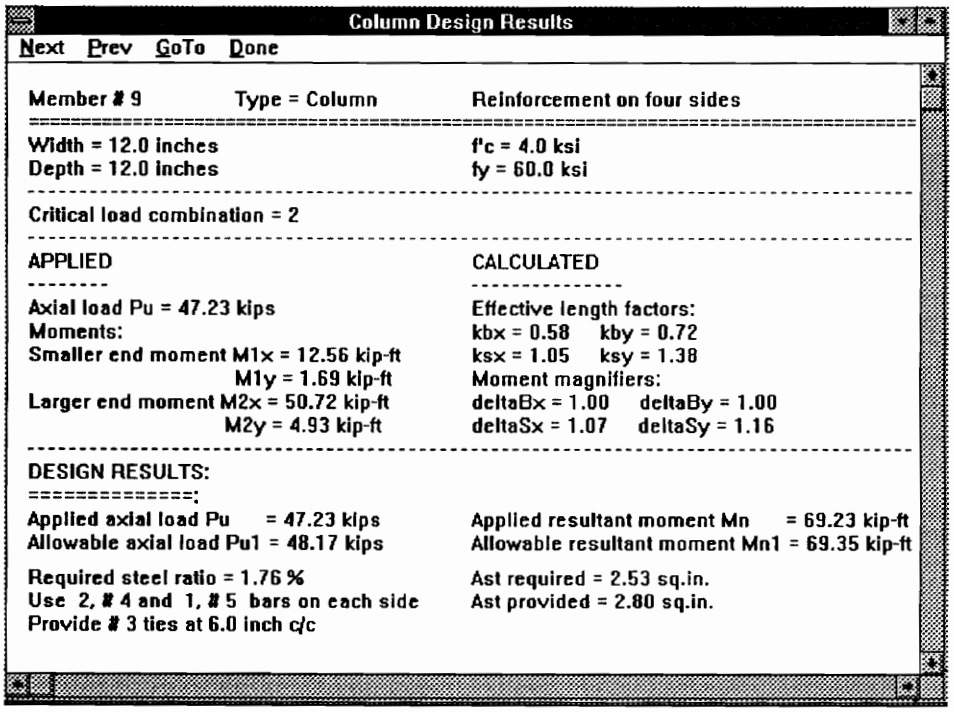


Fig 3.15 Column Design Results

In this chapter, a description of the post processor for the design of reinforced concrete space frames was presented. The procedure used for the design of beams and columns was described. The default design parameters were also presented. The program operation was discussed. Also, the various menus and dialog boxes for obtaining user input were presented along with a description of the output available from the program.

Chapter 4
Program Architecture

4.1 Introduction

The object oriented programming technique was used in developing the post processor. As mentioned in Chapter 2, in object oriented programming, the basic building blocks are classes and objects. In this chapter, the major classes used in developing the program are described. The class hierarchy and data flow relationships are represented with the help of diagrams. In order to help differentiate between objects and classes used in the program and the corresponding objects in a real structure, class and object names are written in a monospaced font. Also the first letter in the class name is capitalized while the name of the object of the class is written in all lower case letters. For example, the class 'Member' is written as `Member` and the 'member' object is written as `member`.

4.2 Overview of Classes

The classes used in the program can be divided into two types: structural classes and interface classes. Structural classes are used to represent objects related to the structural model such as beams and columns. Structural classes are used to store and manipulate data related to the structure. Interface classes are used for representing Windows graphical user interface elements such as windows, dialog boxes and buttons. All classes are defined in such a manner that they represent objects in the real world. For example, the structural object `member` in the program represents a real member of the structure to be designed. Interface classes are derived using the Borland Object Windows Library (OWL).

4.3 Structural Classes

The major structural classes used in the program are: `Structure`, `LoadCase`, `LoadCombination`, `Member`, `Joint`, `Beam`, `Column`, `BeamOutput`, and `ColumnOutput`. The concept of inheritance is not used in defining structural classes in order to avoid the memory overhead associated with inheritance. Instead the concept of class friendship is used for sharing data between structural objects. Fig. 4.1 shows the relationship among the structural classes.

A brief description of these classes is given below. A more complete definition of each class is provided in Appendix A.

4.3.1 `Structure` Class

The class `Structure` contains the general description of the structure to be designed. Data stored in a `structure` object includes: name of the structure, type of structure (i.e., plane frame or space frame), number of members, number of joints, number of beams, number of columns, total number of supports and number of fixed and hinged supports. The `Structure` class has a member function called `getdata` which receives all of the above data. All of the data members of class `Structure` are declared as public and hence are accessible from outside a `structure` object.

4.3.2 `LoadCase` Class

The class `LoadCase` stores information about a load case. An instance of this class is created for each load case. The data stored in an object of this class includes: load case number, the type of load (dead, live, wind, etc.) and the total number of concentrated loads, uniform loads and linear loads in the load case. This class also contains a member

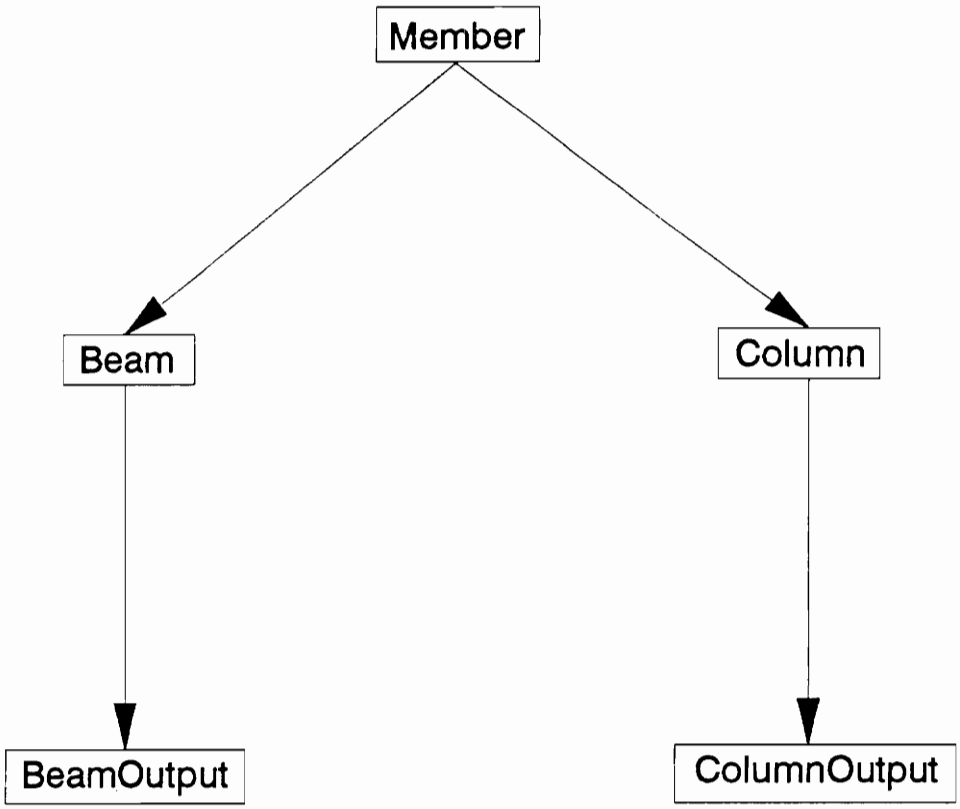


Fig. 4.1 Data Flow Relationship

function called `getdata` that obtains the above data. As with the `Structure` class, all data members are declared as public and are accessible from outside the object.

4.3.3 `LoadCombination` Class

The class `LoadCombination` stores information regarding load combinations. One object of this class is created for each load combination. The data stored in an object of the class `LoadCombination` includes: number of load cases in the load combination, and the load case numbers and corresponding load factors for each load case in the load combination. All data members are declared as public.

4.3.4 `Member` Class

The class `Member` stores information about an individual member such as a beam or a column. One `member` object is created for each member. Thus, each `member` object represents a real member of the structure. The data stored in the `member` object includes: member number, member incidences, member end coordinates, length of the member, type of member (beam or column), material properties (compressive strength of concrete, yield strength of steel, Young's modulus, etc.), cross-section properties (area and moment of inertia), loads acting on the member, and member end forces. All of the above information is stored as private data (accessible only from within the object) except information regarding the type of the member. The type of the member is declared as public.

The `member` object contains objects of the classes `UniLoad`, `PtLoad` and `LinearLoad` as its private data to store information regarding applied loads. An object of class `MemberForce` is used to store member end forces.

The class `Member` defines several functions to receive the data from the input file. Some of the these functions are listed below along with their assigned tasks.

`get_joint_coordi:`

Obtains joint coordinates and determines the type of the member (beam or column) from the joint coordinates and accordingly assigns the default design parameters

`get_section:`

Obtains the section properties (area and moment of inertia) and assuming a rectangular section, computes section dimensions using the area and the moment of inertia

`get_E:`

Computes the modulus of elasticity based on the value of f_c' using the equation: $E = 57000\sqrt{f_c'}$

`get_udl_data:`

Obtains data regarding uniform loads

`get_tri_ld_data:`

Obtains data regarding linear loads

`get_ptld_data:`

Obtains data regarding concentrated loads

`get_lt_end_forces:`

Obtains left end forces

`get_rt_end_forces:`

Obtains right end forces

`get_fc_fy:`

Obtains values for the strength of concrete f_c' and yield strength of steel f_y

`get_StNum:`

Obtains the stirrup size

`get_cover:`

Obtains the cover to the reinforcement

The `Member` class also defines the data and functions required to calculate effective length factors for column. If the member is a column then the function `cal_k` is called to calculate the effective length factors of the column in accordance with the ACI specifications. The calculation of effective length factors involves the solution of Equation (1) for the bending condition and Equation (2) for the sway condition [Salmon and Johnson, 1980].

$$f(p) = 0.25\ \Psi_a\ \Psi_b\ (p)^2 + 0.5\ (\Psi a + \Psi b)\ (1 - p\ \cot(p)) + 2.0\ \tan(0.5p) / p - 1 \quad (1)$$

$$f(p) = \Psi_a\ \Psi_b\ (p)^2 + 6.0\ (\Psi a + \Psi b)\ p\ \cot(p) - 36.0 \quad (2)$$

where:

Ψa = ratio of the sum of the column rotational stiffnesses to the sum of the beam rotational stiffnesses at the first end of the member.

Ψb = ratio of the sum of the column rotational stiffnesses to the sum of the beam rotational stiffnesses at the second end of the member.

k = effective length factor

p = π / k

Equation (1) is solved by calling the function `BrSolve_k` and the Equation (2) is solved by calling the function `NBrSolve_k`. The unsupported lengths for both axes of bending are also calculated while calculating the effective length factors.

The class `Member` also has functions for computing the transformed coordinates of the ends of the member in order to display the member on the screen. The class

Member is declared as a friend of the classes Beam, Column, TBeamParaDlg and TColumnParaDlg, so that private data of the member object is also accessible from within objects of these classes.

The member object reads data regarding the member from the input file (results of the analysis), manipulates its own data, lets the user modify this data through the object TBeamParaDlg (if the member is a beam) or TColumnParaDlg (if the member is a column), and passes information to the corresponding object of the class Beam or Column. TBeamParaDlg and TColumnParaDlg represent dialog boxes that allow the user to change beam and column design parameters respectively.

4.3.4.1 UniLoad Class

The class UniLoad corresponds to a uniformly distributed load applied to a member. Each object of this class represents a uniform load applied on a given member. The data stored in the object of this class includes: magnitude of the applied load and the distances to the start and end of the load. All of the above data are declared as public, but since this object is created as a private data of the member object, the data is only accessible from within a member object.

4.3.4.2 PtLoad Class

The class PtLoad defines information regarding each concentrated load acting on a member. One instance of the class PtLoad is created for each concentrated load. The data members for this class include the magnitude and point of application of the concentrated load. All data members are declared as public. Like an object of the class UniLoad, an object of class PtLoad is also created as a private data of the member object, and hence the data is only accessible from within the member object.

4.3.4.3 `LinearLoad` Class

The class `LinearLoad` stores information regarding each linear load acting on a member. The data stored in the object of this class includes: load intensity at the start and end of the load and distances to the start and end of the load. All data members are declared as public. An object of the class `LinearLoad` is also created as a private data member of the `member` object.

4.3.4.4 `MemberForce` Class

The class `MemberForce` represents member end forces. Thus, each object of this class represents the calculated member end forces for each load case. The data stored in an object of this class includes: axial force, shear forces (in the local y and z directions), bending moments about local y and z directions and the torsional moment. All data is declared as public and the object itself is created as the private data of the `member` object.

4.3.5 `Joint` Class

The class `Joint` stores information regarding an individual joint in the structure. One `joint` object is created for each joint in the structure. The data stored in the `joint` object includes: joint number, joint coordinates, type of joint (free or support), joint loads and joint displacements. The `Joint` class contains data members of type `JointLoad` for sharing information regarding applied joint loads. It also contains data members of class `Displacement` for storing joint displacements. All of the above data is stored as private data. The class `Joint` is declared as a friend of the classes `Beam` and `Member`. The class `Joint` also defines functions to receive and manipulate its data. For example, the function `get_type` is called to receive data regarding joint constraints. The function

getdata obtains the joint coordinates and the function get_displacements is called to obtain joint displacements.

4.3.5.1 Displacement Class

The class Displacement stores joint displacements. One object of this class is created for each joint and load case. The data stored includes: rotations and displacements in the global x, y and z axes. All data is declared as public. This object is created as the private data of the joint object, and hence the data is only accessible from within the joint object.

4.3.5.2 JointLoad Class

The class JointLoad stores applied joint loads. One object is created for each joint load acting at a joint. The data stored in the object of this class includes: magnitude and direction of the load and the load case to which this load belongs. All data is declared as public. Since this object is created as the private data of the joint object, the data is only accessible from within the joint object.

4.3.6 Beam Class

The class Beam contains the data members and functions necessary to design an individual beam. The beam object represents an actual beam in the structure. All data members of this class are private while all member functions are public. When a beam object is created, the function get_memb_data is called to receive the data from the member object. The data received from the member object includes: member number, cross-sectional properties, material properties, loading on the beam and member end forces. All of the functions required to design a beam are defined in class Beam. The

function `analyze` is called to compute shear forces and bending moments at fifteen equally spaced sections along the beam. The function `design_forces` determines the maximum design shear force and the maximum positive and negative bending moment at each section considering all the load combinations. The function `Maximum` determines the absolute maximum bending moment acting on the beam. The function `steel_ratios` computes the balanced steel ratio, the maximum and minimum steel ratios and the design steel ratio. If the option to select the new dimensions has been specified, the function `Findbd` is called for computing the set of suitable widths and depths for the beam. `Findbd` is also called if the given section dimensions are inadequate and the section has to be changed for the following reasons; a) the shear capacity of the section is inadequate, or b) the moment capacity is inadequate for a singly reinforced section but the user has indicated that a doubly reinforced section is not to be used. The function `IsSingly` checks whether the section can be designed as the singly reinforced section. If the section can be designed as the singly reinforced section, the function `Find_Astx` is called to compute the area of tension reinforcement at each of the fifteen sections. When the beam section has to be designed as a doubly reinforced section, the function `Find_Asx` is called to compute the required area of tension reinforcement as well as compression reinforcement at each section. The function `shearDesign` computes the spacing of stirrups at each section.

The function `find_max_steel` is called when the beam belongs to a mark group and the user has specified that the same size and reinforcement to be used for all beams in the mark group. This function compares the computed areas of longitudinal steel for all beams in the mark group and determines the maximum values. These values are then used for all the beams in the mark group.

Class Beam is defined as a friend of class BeamOutput. The beam object receives the required data from the corresponding member object, designs the beam and passes the design results to the corresponding beamOutput object.

4.3.7 BeamOutput Class

Class BeamOutput stores the results of the beam design. One object of this class is created for each beam in the structure. All data stored in this object is declared as public. Data stored in this object includes: member number, section and material properties, group number (in the case of mark groups), size of the stirrups, design shear forces, positive and negative design moments, area of longitudinal steel required at top and bottom and spacing of stirrups at fifteen sections. When the design of the beam is completed, a beamOutput object is created and its function get_data is called to receive the above data from the corresponding beam object. Since class Beam is defined as a friend of the class BeamOutput, the private data members of the beam object are accessible to objects of class BeamOutput.

4.3.8 Column Class

The class Column stores the data and the functions necessary to design a column. When a column object is created, the Column constructor is called which obtains the necessary data from the corresponding member object. The data received from the member object includes: member number, section properties, material properties, member end forces, effective length factors and unsupported length. All of the functions required to design a column are defined in the class Column. The function design is called to design the longitudinal steel for the column. If the column is a slender column then the function Magnify_b is called to magnify the moments due to bending and the function

Magnify_s is called to magnify the moments due to sway. If the column is subjected to biaxial bending, the column is designed for the resultant uniaxial moment and the function Reciprocal is called to check the adequacy of the design using the reciprocal load method.

The column is designed for each load combination. The function find_steel computes the area of longitudinal steel for each load combination. The function pick_critical is called to determine the critical load combination. The load combination requiring the largest longitudinal steel area is considered as the critical load combination. The function find_bars is called to determine the possible bar combinations for the critical load combination. This function considers several bar combinations for bar sizes ranging from # 4 to # 18 that give an area of steel between 100% and 140% of the required steel area. A maximum of two bar sizes is considered. The bars cannot be more than two sizes apart. Also, a maximum of five bars of each size in the case of two different bar sizes or a maximum of ten bars in the case of a single bar size is considered. The function find_bars calls the function check_width to check the adequacy of the section dimensions to accommodate the chosen bars. It also calls the function check_symmetry to ensure that the chosen bars can be placed symmetrically in the section. The function pick_most_ecconomical is called to select the final bar combination from the set of possible bar combinations. The function selects the single bar size arrangement if the total area of steel for the single bar size is within five percent of the required steel area, otherwise the bar combination with the least area of steel is chosen. Finally the function design_ties is called to design the lateral ties for the column.

The Class Column is defined as a friend of class ColumnOutput. The column object receives the required data from the corresponding member object, designs the column and passes the design results to the corresponding columnOutput object.

4.3.9 `ColumnOutput` Class

Class `ColumnOutput` stores the results of the column design. An instance of this class is created for each column. All data members are declared as public. Data stored in the object of this class includes: member number, section and material properties, critical axial force, critical moments about major and minor axes, critical resultant uniaxial moment, calculated effective length factors and moment magnifiers, allowable axial force and resultant moment and longitudinal steel and tie requirements. When the design of a column is completed, a `columnOutput` object is created to store the design results. The function `get_data` obtains the data from the corresponding `column` object.

4.4 Interface Classes

Interface classes are used for defining user interface elements such as windows and dialog boxes used in the Windows environment. The concept of inheritance is used in defining all interface classes. All interface classes are derived using the Borland Object Windows Library. Table 4.1 lists the interface classes used in the program. Also shown in Table 4.1 are the base classes and the tasks performed by the classes. The classes defining windows are derived from the OWL class `TWindow`. The classes for implementing the File Open and File Save dialog boxes are derived from the class `TFileDialog`. All other classes for dialog boxes are derived from the class `TDialog`. A complete definition of these classes can be found in Appendix A. Details of the Borland Object Windows

Table 4.1 Interface Classes

Interface Class	Base Class	Purpose
TPostproWindow	TWindow	Main window of the application.
TBeamResultsChild	TWindow	Child window used to display beam design results.
TColumnResultsChild	TWindow	Child window used to display column design results.
TColumnDesignChild	TWindow	Child window that informs the user about the progress of the column design.
TFileDlg	TFileDialog	Implements the File Open and File Save As dialog boxes.
THelpDlg	TDialog	Help About dialog box to provide the information about the program.
TLoadCaseDlg	TDialog	Load Case dialog box to allow the user to specify load cases.
TBeamMarkGroupDlg	TDialog	Mark Group dialog box to allow the user to assign mark groups to beams.
TBeamParaDlg	TDialog	Beam Parameters dialog box for modifying default design parameters for beam design.
TColumnParaDlg	TDialog	Column Parameters dialog box for modifying default design parameters for column design.
TSectionDlg	TDialog	Section Selection dialog box for selecting section dimensions for beams.
TInfDblyDlg	TDialog	Doubly Reinforced Section message box. Inform user that section has to be designed as doubly reinforced section.
TBeamMembNumDlg	TDialog	Beam Member Number dialog box for specifying the member number of the beam for which design results are required.
TColumnMembNumDlg	TDialog	Column Member Number dialog box for specifying the member number of the column for which the design results are required.

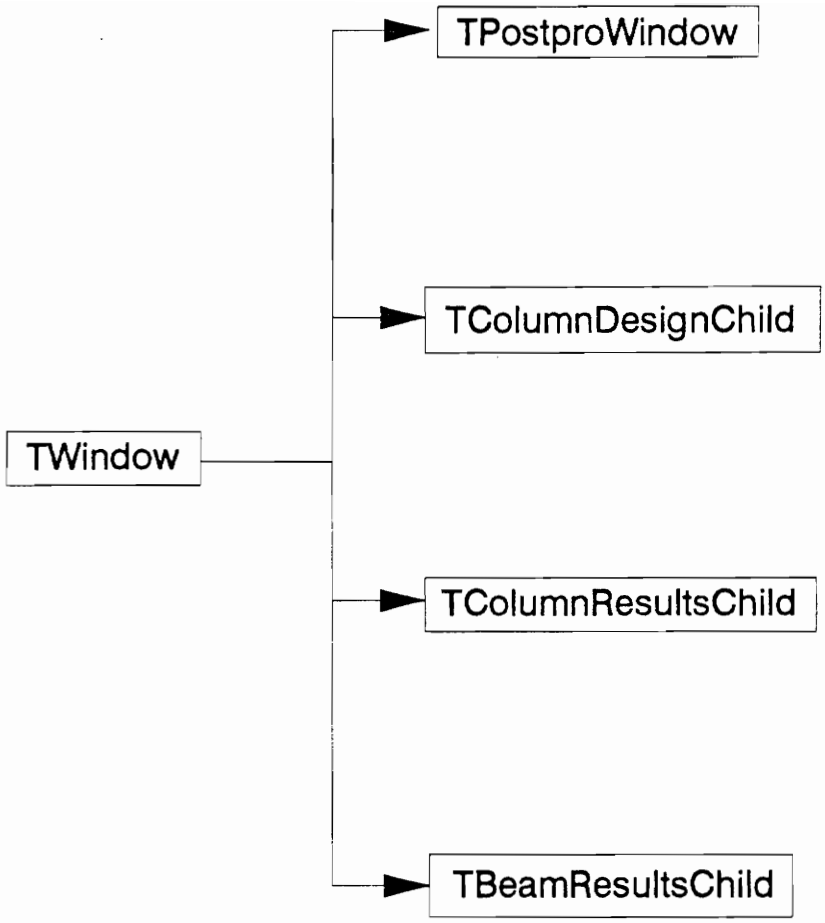


Fig. 4.2 Interface Classes Derived from `TWindow`

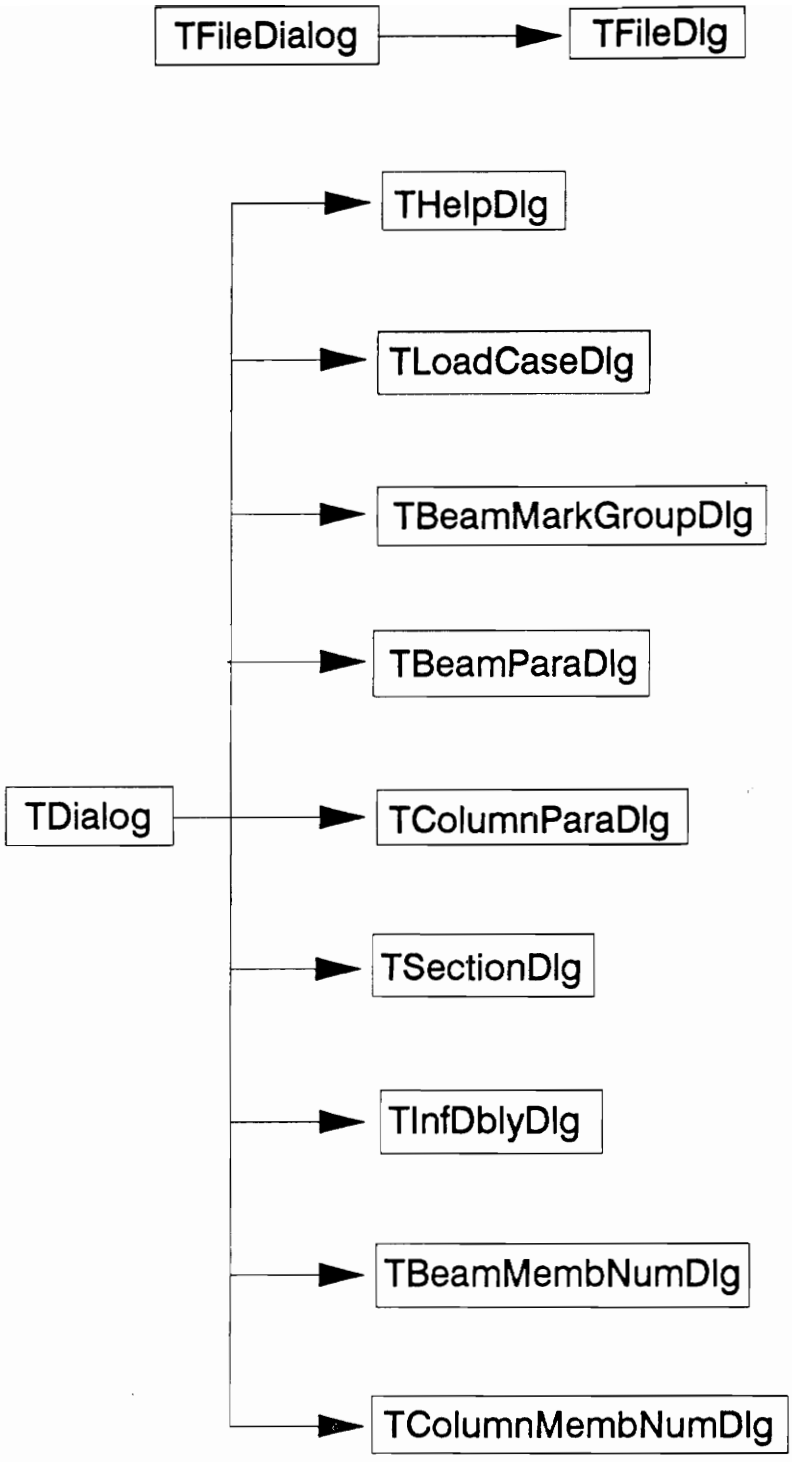


Fig 4.3 Interface Classes Derived from `TFileDialog` and `TDialog`

Library classes are available in the Borland C++ Programming Manual [Borland Inc., 1992]. Figures 4.2 and 4.3 show the hierarchy of the interface classes.

In this chapter, the classes used in the program were described. There are two types of classes in the program: structural classes and interface classes. The structural classes were described. Also, the interface classes were described and the class hierarchy was presented.

Chapter 5
Program Results and Discussions

5.1 Introduction

In this chapter, a comparison of results obtained using the post processor and those obtained from commercial programs is presented. Two structures were designed using the post processor. The first structure was a four span continuous beam and the second was a two story space frame. The results of the continuous beam design were compared with those obtained from Research Engineer's STAAD III program [Research Engineers, Inc.] and Intergraph's MicasPlus program [Intergraph Corporation]. The results obtained using the post processor for the design of the space frame were compared with those obtained using STAAD III.

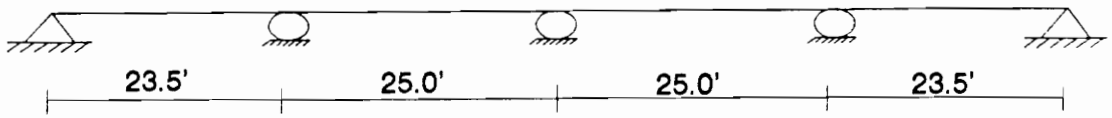
5.2 Continuous Beam

5.2.1 General Description

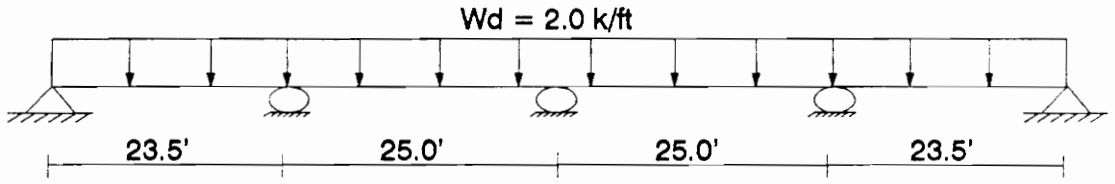
A four span continuous beam was designed using the post processor. The beam had a rectangular cross section of size 18 x 26 inch. The exterior spans were 23.5 feet long while interior spans were 25 feet long. The exterior ends were pinned while all interior supports were roller supports. The concrete compressive strength was 4 ksi and yield strength of steel was 60 ksi. Fig. 5.1(a) shows the beam geometry.

5.2.2 Loading

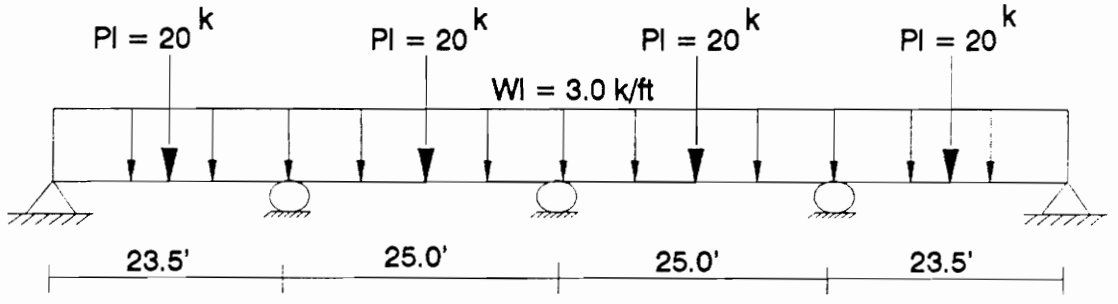
The loading on the beam consisted of two load cases: dead load and live load. The dead load consisted of a uniform load of 2 kips/ft plus the self weight of the beam. Fig.



(a) Beam Geometry



(b) Dead Load



(c) Live Load

Fig 5.1 Continuous Beam

5.1(b) shows the dead load. The live load consisted of a uniform load of 3 kips/ft and concentrated loads of 20 kips acting at the center of each span. Fig. 5.1(c) shows the live load. The beam was designed for the load combination of 1.4 times the dead load plus 1.7 times the live load.

5.2.3 Input and Output Files

Appendix B shows the input data file (which is output file from CADKEY frame analysis program) and the output file of the post processor containing design results.

5.2.4 Comparison of Design Results

The same structure was analyzed and designed by Betaque [Betaque, 1993]. In the following sections, the design results obtained from the post processor are compared with those obtained by Betaque using STAAD III and MicasPlus.

5.2.4.1 Longitudinal Reinforcement

Table 5.1 shows the comparison of design results for longitudinal reinforcement for the beams obtained using the post processor and STAAD III. The maximum difference in the results for longitudinal steel was 1.27%. Thus, for practical purposes, the results obtained from the post processor and STAAD III were the same.

Table 5.2 shows the comparison of design results for longitudinal reinforcement using the post processor and MicasPlus. The differences in the longitudinal steel areas were less than 3.21% except at the right end of the second beam. The difference at this section was 9%. At the same section, Betaque found a difference of 7.83% for the longitudinal steel areas computed by STAAD III and MicasPlus. This was because of the reason that "MicasPlus increments the determined steel area to the area of the next

Table 5.1 Comparison of longitudinal steel area with STAAD III

a) Area of steel at top

Element Number	Post Processor		STAAD III		Difference	
	Left end (sq. in.)	Right end (sq. in.)	Left end (sq. in.)	Right end (sq. in.)	Left end (%)	Right end (%)
1	-	7.55	-	7.62	-	0.93
2	7.55	5.46	7.62	5.53	0.93	1.27
3	5.46	7.55	5.53	7.62	1.27	0.93
4	7.55	-	7.62	-	1.27	-

b) Area of steel at bottom

Element Number	Post Processor	STAAD III	Difference
	Midspan (sq. in.)	Midspan (sq. in.)	Midspan (%)
1	4.95	5.00	1.00
2	3.06	3.08	0.65
3	3.06	3.08	0.65
4	4.95	5.00	1.00

Table 5.2 Comparison of longitudinal steel area with MicasPlus

a) Area of steel at top

Element Number	Post Processor		MicasPlus		Difference	
	Left end (sq. in.)	Right end (sq. in.)	Left end (sq. in.)	Right end (sq. in.)	Left end (%)	Right end (%)
1.00	-	7.55	-	7.80	-	3.21
2.00	7.55	5.46	7.80	6.00	3.21	9.00
3.00	5.46	7.55	6.00	7.80	9.00	3.21
4.00	7.55	-	7.80	-	3.21	-

b) Area of steel at bottom

Element Number	Post Processor	MicasPlus	Difference
	Midspan (sq. in.)	Midspan (sq. in.)	Midspan (%)
1	4.95	5.00	1.00
2	3.06	3.08	0.65
3	3.06	3.08	0.65
4	4.95	5.00	1.00

possible bar combination" [Betaque, 1993]. Thus, for practical purposes, the results obtained from the post processor and MicasPlus were considerably closer than 9% even at this section.

5.2.4.2 Shear Reinforcement

Table 5.3 shows the comparison of the design results obtained for stirrup spacing using the post processor, MicasPlus and STAAD III. The stirrup spacing computed by the post processor agreed very closely to those computed by both MicasPlus and STAAD III. At any given section, the difference in the stirrup spacing computed by all three programs were within 0.1 inch. Thus, for all practical purposes, the results obtained from the post processor were the same as those obtained using STAAD III and MicasPlus.

5.3 Two Story Frame

5.3.1 General Description

Figure 5.2 shows the two story space frame designed using the post processor. The first story was 12 feet high and second story was 10 feet high. The frame had two 30 feet by 30 feet bays. Thus, the total length of the structure was 60 feet and the total width of the structure was 30 feet. All supports were fixed supports and all other joints were rigid. The frame considered was a moment resisting frame. The effective length factors and the moment magnifiers were all set equal to one. Thus, all the columns were treated as short columns. The compressive strength of concrete was 4 ksi and the yield strength of steel was 60 ksi.

Table 5.3 Comparison of stirrup spacing with STAAD III and MicasPlus

a) Comparison of stirrup spacing with MicasPlus

Element Number	Post Processor		MicasPlus		Difference	
	Left end (in.)	Right end (in.)	Left end (in.)	Right end (in.)	Left End (%)	Right End (%)
1	11.8	5.6	11.7	5.6	0.85	0.00
2	6.9	8.5	6.9	8.5	0.00	0.00
3	8.5	6.9	8.5	6.9	0.00	0.00
4	5.6	11.8	5.6	11.7	0.00	0.85

b) Comparison of stirrup spacing with STAAD III

Element Number	Post Processor		STAAD III		Difference	
	Left end (in.)	Right end (in.)	Left end (in.)	Right end (in.)	Left End (%)	Right End (%)
1	11.8	5.6	11.8	5.7	0.00	1.78
2	6.9	8.5	7.0	8.6	1.45	1.18
3	8.5	6.9	8.6	7.0	1.18	1.45
4	5.6	11.8	5.7	11.8	1.78	0.00

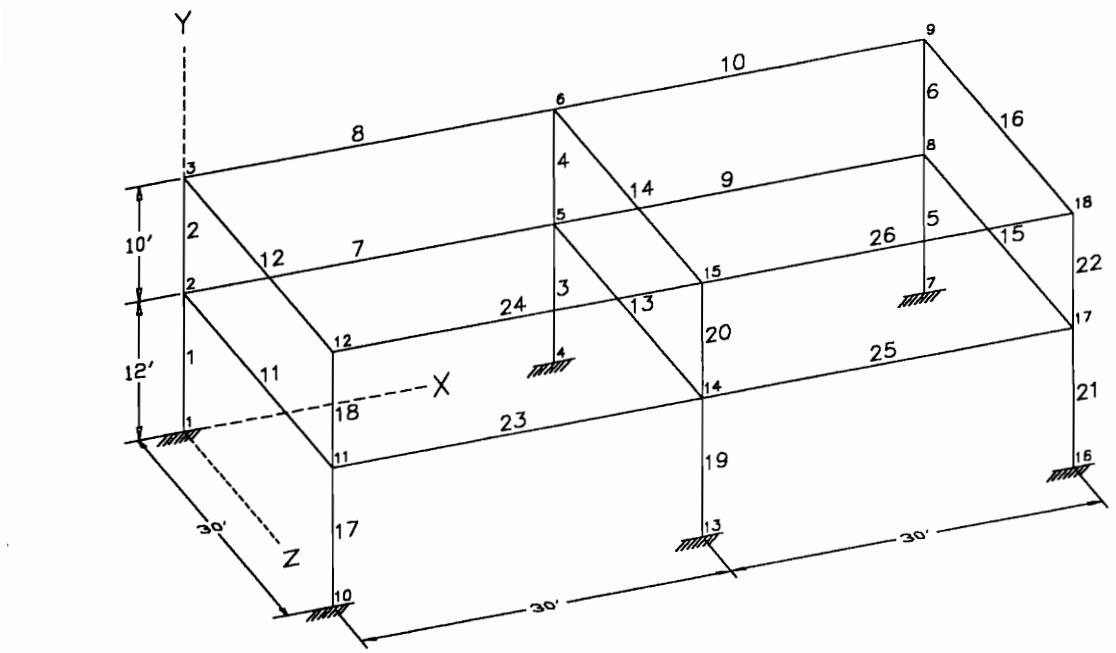


Fig. 5.2 Two Story Space Frame

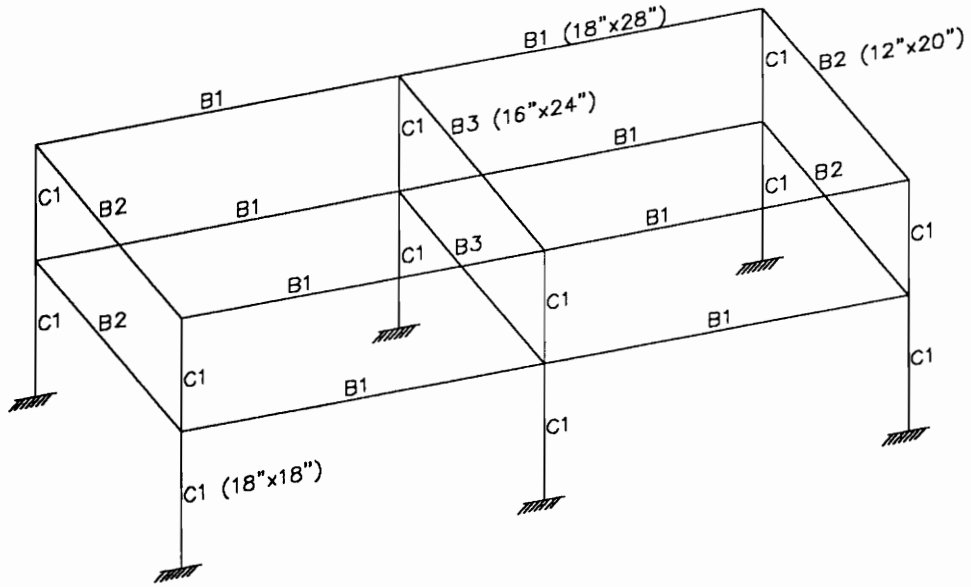


Fig 5.3 Member Nomenclature and Dimensions

Fig. 5.2 shows the member numbers. Fig. 5.3 shows the cross section dimensions of the members. Beams B1 were 18"x18", beams B2 were 12"x20" and beams B3 were 16"x24". All columns were 18"x18".

5.3.2 Loading

Area loads of 50 psf live load and 120 psf dead load were applied to the floor and roof. The resultant forces on the beams were determined. Figures 5.4 and 5.5 show the dead loads and the live loads acting on the frame. The two concentrated loads on beam B1 are the support reactions from secondary beams. The secondary beams were not considered in the analysis and design. To account for the self weight of the secondary beams, an additional dead load of 5.5 kips was applied to beams B1 at each of these concentrated load locations. A wind load of 25 psf was applied to the left side wall as well as to the front wall. Using appropriate tributary areas, the wind loads were discretized into equivalent joint loads. Figures 5.6 and 5.7 show these equivalent joint loads.

The following four load combinations were considered in the analysis and design.

1.4D + 1.7L

0.75(1.4D + 1.7L + 1.7W(left))

0.75(1.4D + 1.7L + 1.7W(front))

0.9D + 1.3W(front)

where:

D = Dead Load

L = Live Load

W = Wind Load

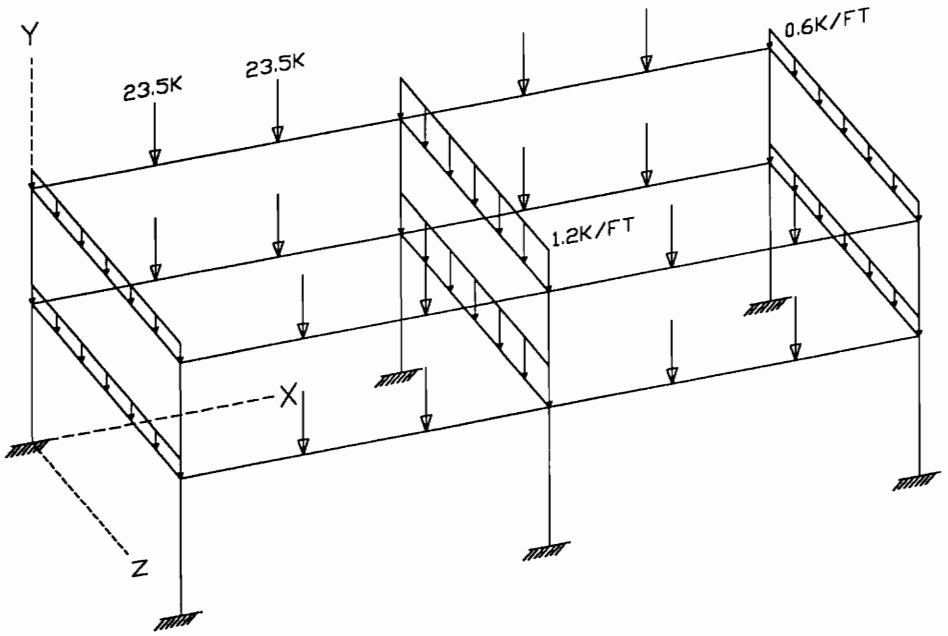


Fig. 5.4 Dead Loads on Frame

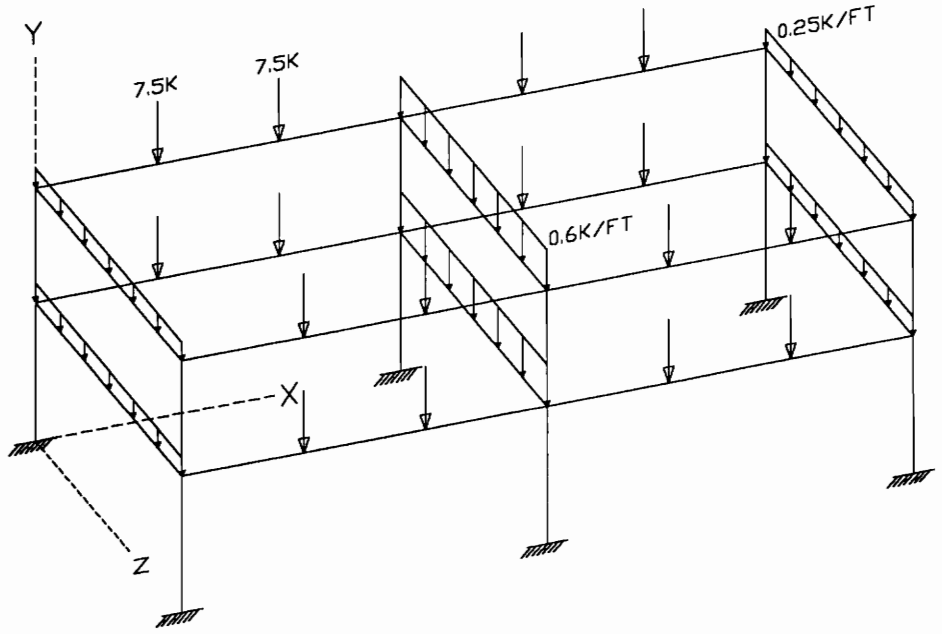


Fig. 5.5 Live Loads on Frame

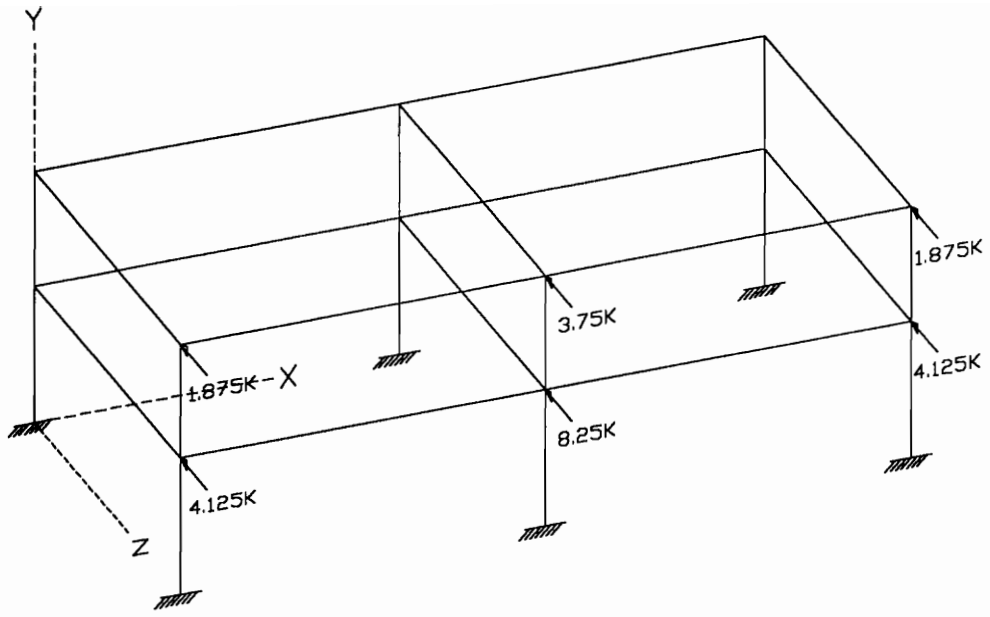


Fig. 5.6 Wind Loads (Front) on Frame

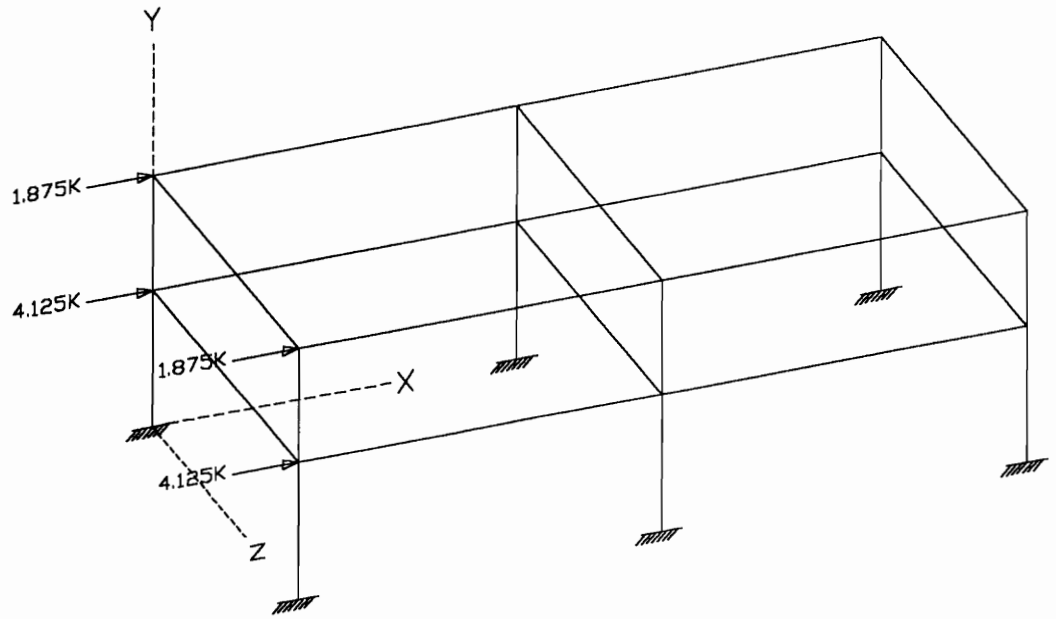


Fig. 5.7 Wind Loads (Side) on Frame

5.3.3 Input and Output Files

Appendix B shows the input data file and the output file from the post processor.

5.3.4 Comparison of Design Results

As mentioned earlier, the same structure was also analyzed and designed using STAAD III. In the following sections, design results obtained from the post processor are compared with those obtained from STAAD III.

5.3.4.1 Beam Design Results

5.3.4.1.1 Longitudinal Reinforcement

Table 5.4 shows the comparison of flexural design results for all beams, obtained using the post processor and STAAD III. When determining required longitudinal steel area, STAAD III does not increase the calculated area of steel to the minimum area of steel required by the ACI specification. The post processor however, increases the calculated steel area to the minimum required area, if the calculated longitudinal area of steel at any section is less than the minimum area. In table 5.4, the percentage difference between the results obtained from the post processor and those obtained from STAAD III is not shown for those cases where the minimum area of steel is required. The maximum percentage difference in longitudinal steel area between the two programs was 1.27%. Thus, the results obtained from the post processor and STAAD III were the same for flexural design of beams.

Table 5.4 Comparison of longitudinal steel area with STAAD III

a) Area of steel at top

Element Number	Post Processor		STAAD III		Difference	
	Left end (sq. in.)	Right end (sq. in.)	Left end (sq. in.)	Right end (sq. in.)	Left end (%)	Right end (%)
7	1.87	3.23	1.87	3.24	0.00	0.31
8	1.53 *	3.46	1.40	3.47	-	0.29
9	3.23	1.87	3.24	1.87	0.31	0.00
10	3.46	1.53 *	3.47	1.40	0.29	-
11	1.20	1.20	1.21	1.21	0.83	0.83
12	1.10	1.10	1.10	1.10	0.00	0.00
13	1.95	1.95	1.96	1.96	0.51	0.51
14	1.71	1.71	1.71	1.71	0.00	0.00
15	1.20	1.20	1.21	1.21	0.83	0.83
16	1.10	1.10	1.10	1.10	0.00	0.00
23	1.87	3.23	1.87	3.24	0.00	0.31
24	1.53 *	3.46	1.40	3.47	-	0.29
25	3.23	1.87	3.24	1.87	0.31	0.00
26	3.46	1.53*	3.47	1.40	0.29	-

* Minimum area of steel required

b) Area of steel at bottom

Element Number	Post Processor	STAAD III	Difference
	Maximum (sq. in.)	Maximum (sq. in.)	Maximum (%)
7	1.80	1.80	0.00
8	2.03	2.04	0.49
9	1.80	1.80	0.00
10	1.89	1.89	0.00
11	0.70 *	0.69	-
12	0.78	0.79	1.27
13	1.33	1.33	0.00
14	1.57	1.57	0.00
15	0.70 *	0.69	-
16	0.78	0.79	1.27
23	1.8	1.8	0.00
24	2.03	2.04	0.49
25	1.8	1.8	0.00
26	1.89	1.89	0.00

* Minimum area of steel required

5.3.4.1.2 Shear Reinforcement

Table 5.5 shows the comparison of the results for shear design of beams obtained from the post processor and STAAD III. The stirrup spacings computed by the post processor were identical to those computed by STAAD III for all beams.

5.3.4.2 Column Design Results

The column design results obtained from the post processor were compared with those obtained using STAAD III. Unlike the post processor, STAAD III does not compute effective length factors or moment magnification factors. Therefore, all of the columns were treated as short columns by setting the values of all moment magnifiers to one. Also, STAAD III always designs the columns with the reinforcement equally distributed on all four faces while the post processor can design columns with reinforcement on either two sides or on all four sides. In order to provide a fair comparison, all columns were designed with reinforcement on all four sides. Each column was designed for all load combinations using the interactive design feature of STAAD III. Table 5.6 shows the area of longitudinal steel computed by both programs for the critical load combination.

The required steel areas obtained from both programs were identical for columns 1, 3, 4, 5, 17, 19, 20 and 21. For columns 2, 6, 18 and 22, the required steel areas obtained using the post processor were 1.21% less than those obtained from STAAD III. For all of the columns, the post processor gave an area of steel that was less than that given by STAAD III. The difference in results between the two programs is due to the fact that the post processor allows for a bar combination containing two different bar sizes, while STAAD III only allows one bar size. Thus, the post processor provides less steel as compared to STAAD III.

Table 5.5 Comparison of stirrup spacing

Element Number	Post Processor		STAAD III		Difference	
	Left end (in.)	Right end (in.)	Left end (in.)	Right end (in.)	Left End (%)	Right End (%)
7	12.8	12.8	12.8	12.8	0.00	0.00
8	12.8	12.8	12.8	12.8	0.00	0.00
9	12.8	12.8	12.8	12.8	0.00	0.00
10	12.8	12.8	12.8	12.8	0.00	0.00
11	8.8	8.8	8.8	8.8	0.00	0.00
12	8.8	8.8	8.8	8.8	0.00	0.00
13	10.8	10.8	10.8	10.8	0.00	0.00
14	10.8	10.8	10.8	10.8	0.00	0.00
15	8.8	8.8	8.8	8.8	0.00	0.00
16	8.8	8.8	8.8	8.8	0.00	0.00
23	12.8	12.8	12.8	12.8	0.00	0.00
24	12.8	12.8	12.8	12.8	0.00	0.00
25	12.8	12.8	12.8	12.8	0.00	0.00
26	12.8	12.8	12.8	12.8	0.00	0.00

Table 5.6 Comparison of column design results with STAAD III

Area of longitudinal steel for columns

Element Number	Post Processor		STAAD III		Difference	
	Required (sq. in.)	Provided (sq. in.)	Required (sq. in.)	Provided (sq. in.)	Required (%)	Provided (%)
1	3.24	3.34	3.24	3.52	0.00	5.39
2	6.61	6.68	6.69	7.04	1.21	5.39
3	3.24	3.34	3.24	3.52	0.00	5.39
4	3.24	3.34	3.24	3.52	0.00	5.39
5	3.24	3.34	3.24	3.52	0.00	5.39
6	6.61	6.68	6.69	7.04	1.21	5.39
17	3.24	3.34	3.24	3.52	0.00	5.39
18	6.61	6.68	6.69	7.04	1.21	5.39
19	3.24	3.34	3.24	3.52	0.00	5.39
20	3.24	3.34	3.24	3.52	0.00	5.39
21	3.24	3.34	3.24	3.52	0.00	5.39
22	6.61	6.68	6.69	7.04	1.21	5.39

In this chapter, design results for two structures obtained using the post processor were compared with those obtained from STAAD III and MicasPlus. The design results for the continuous beam obtained using the post processor were compared with those obtained from STAAD III and MicasPlus. The results obtained using the post processor for the design of space frame were compared with those obtained from STAAD III.

Chapter 6
Summary and Conclusions

The objective of this thesis was to develop a Windows based post processor for the design of reinforced concrete space frames. In order to provide an integrated application for analysis and design, the post processor reads in the analysis results from the output file of the CADKEY frame analysis program. The post processor designs the beams and columns of plane and space frames in accordance with ACI specifications. The post processor can perform both flexural and shear design of beams. In the flexural design of beams, the negative and/or positive reinforcement at fifteen equally spaced sections along the beam is determined. In shear design, the stirrup spacing at these sections is determined. Beams can be assigned to a mark group thus making it possible to design beams having similar section dimensions and/or reinforcement. The beam design process is interactive and allows the engineer considerable control over the choice of beam dimensions and the type of beam (singly reinforced or doubly reinforced).

The post processor is capable of designing rectangular tied columns. The reinforcement can be placed on all four sides or on two sides of the column. The post processor also has the capacity for computing the effective length factors and the moment magnification factors for long columns. The post processor can determine the longitudinal reinforcement required and offer suggestions regarding bar combinations. While selecting bar combinations the program is capable of considering combinations of bars of two different sizes. The program also designs the lateral reinforcement (ties) for the columns.

The post processor was developed using the object oriented programming approach. Object oriented techniques were used for both the structural design aspects as well as the user interface aspects of the program. It was observed that the use of object oriented programming has many advantages. The objects in the program represent objects in the real world such as beams and columns. The one to one analogy with the real world made it possible to develop a program which is easy to understand. The use of object oriented techniques also resulted in a program which was easier to develop and debug. The use of Borland's OWL (Object Windows Library) library of classes greatly reduced the effort required for developing the program for the Windows graphical user interface.

The program demonstrates the use of object oriented programming techniques for developing a structural design application. The post processor is only capable of designing reinforced concrete members. However, it can easily be extended for the design of steel frames. It can also be modified for the design of reinforced concrete space frames according to other design codes and specifications. The frame work developed in the program represents one possible way to apply object oriented techniques to develop structural design applications and does not necessarily represent the best possible way. Hence, other possible ways should also be investigated in future.

References

Abdalla, J. A. and Yoon, C. J., "Object-Oriented Finite Element and Graphics Data-Translation Facility", Journal of Computing in Civil Engineering, Vol. 6, No. 3, July 1992.

Adeli, H. and Yu, G., "Computer Aided Design of Structures", Microcomputers in Civil Engineering, Vol. 6, No. 3, 1991.

Akhras, G. and Foo, H. C., "Building Object-Oriented Systems for Diagnostic Problems in Civil Engineering", Computing in Civil and Building Engineering, Proceedings, International Conference on Computing in Civil and Building Engineering, 1993.

Atkinson, L. and Atkinson, M., "Using Borland C++", QUE Corporation, 1991.

Baugh, J. W. Jr. and Rehak, D. R., "Object-Oriented Design of Finite Element Programs", Computer Utilization in Structural Engineering, Proceedings of the Sessions Related to Computer Utilization at ASCE Structures Congress '89, 1989.

Betaque, A. D. and Rojiani, K. B., "Evaluation of Software for Analysis and Design of Reinforced Concrete Structures", Proceedings of the ASCE First Congress on Computing in Civil Engineering, Washington DC, June 20-22, 1994 (to be published).

Betaque, A. D., "Evaluation of Software for Analysis and Design of Reinforced Concrete Structures", M.S. Thesis, Virginia Polytechnic Institute and State University, 1993.

Biedermann, J. D. and Grierson, D. E., "Computer-based Design of Civil Engineering Structures Using Object-Oriented Programming", Applications of Artificial Intelligence in Engineering, 1992.

"Borland C++ Programming Manual", Borland Inc., Scott Valley, CA, 1992.

"Building Code Requirements for Reinforced Concrete (ACI 318-89) and Commentary (ACI 318R-89)", American Concrete Institute, Detroit, Michigan, 1990.

Dubois-Pelerin, Y., Zimnermann, T. and Bomme, P., "Object-Oriented Finite Element Programming Concepts", Computer Methods in Applied Mechanics and Engineering, Vol. 98, No. 2, 1992.

Ezzel, B., "Turbo C++ Programming", Addison-Wesley Publishing Company, Inc., 1990.

Forde, B. W. R., Foschi, R. O. and Stiemer, S. F., "Object Oriented Finite Element Analysis", Computers and Structures, Vol. 34, No. 3, 1990.

Garrett, J. H. Jr., "An Object-Oriented Representation of Design Standards", Computing in Civil Engineering, Proceedings of the Sixth Conference, September 11-13, Atlanta, 1989.

Hakim, M. and Garrett, J. H., "Object-Oriented Techniques for Representing Engineering Knowledge and Data", Application of Artificial Intelligence in Engineering, 1992.

Huges, J. G., "Object Oriented Databases", Prentice Hall International Series in Computer Science, New York, 1991.

Jaiswal, S. S. and Riggs, H. R., "Design of An Object-Oriented Structural Analysis Program", Proceedings of First International Offshore Polar Engineering Conference, 1991.

Kulkarni, A. B. and Rojiani, K. B., "An Object-Oriented Approach for Reinforced Concrete Design", Proceedings of the ASCE First Congress on Computing in Civil Engineering, Washington DC., June 20-22, 1994 (to be published).

Kulkarni, A. B.,"Object Oriented Programming for Reinforced Concrete Design", M.S. Thesis, Virginia Polytechnic and State University, 1993.

Lafore, R., "Object Oriented Programming in Turbo C++", Waite Group Press, Mill Valley, CA, 1991.

Lee, H. H. and Arora, J. S., "Object-Oriented Programming for Engineering Applications", Engineering with Computers, Vol. 7, No. 4, 1991.

"MicasPlus Concrete Design Reference Guide", Intergraph Corporation, One Madison International Park, Huntsville, Albama.

McCord, J. W., "Windows Programmer's Guide to Borland C++ Tools", SAMS, 1992.

Micallef, J., "Encapsulation, Reusability and Extensibility in Object-Oriented Programming Languages", Journal of Object-Oriented Programming, April / May 1988.

Miller, G. R., "A LISP-Based Object-Oriented Approach to Structural Analysis", Engineering with Computers, Vol. 4, No. 4, 1988.

Miller, G. R., "An Object Oriented Approach to Structural Analysis and Design", Computers and Structures, Vol. 40, No. 1, 1991.

Miller, G. R., "Object Oriented, Concurrent Structural Analysis", Computing in Civil Engineering, Proceedings of the Sixth Conference, September 11-13, Atlanta, 1989.

Powell, G. H. and Abdalla, G. A., "Object Management in an Integrated Design System", Computing in Civil Engineering, Proceedings of the Sixth Conference, September 11-13, Atlanta, 1989.

Powell, G. H. and An-Nashif, A. N., "An Object-Oriented Algorithm for Automated Modelling of Frame Structures: Stiffness Modelling", Engineering with Computers, Vol 7, No. 2, 1991.

Powell, G. H. and Bhateja, R., "Data Base Design for Computer-Integrated Structural Engineering", Engineering with Computers, Vol. 4, No. 3, 1988.

Powell, G. H., Abdalla, G. A. and Sause, R., "Object Oriented Knowledge Representations: Cute Things and Caveats", Computing in Civil Engineering, Proceedings of the Sixth Conference, September 11-13, Atlanta, 1989.

Remy, P., Devloo, B. and Filho, J. S. R. A., "An Object Oriented Approach to Finite Element Programming (Phase I): A System Independent Windowing Environment for Developing Interactive Scientific Programs", Advances in Engineering Software, Vol. 14, No. 1, 1992.

Roetzheim, W., "Programming Windows with Borland C++", Ziff-Davis Press, Emeryville, California, 1992.

STAAD III / ISDS Users Manual; Release 12, Research Engineers, Inc., Orange, California.

Tello, E. R., "Object Oriented Programming for Windows", John Willey & Sons, Inc., 1991.

Yu, G. and Adeli, H., "Computer-Aided Design Using Object-Oriented Programming Paradigm and Blackboard Architecture", Microcomputers in Civil Engineering, Vol. 6, 1991.

Appendix A
Class Definitions for the Post Processor

In this Appendix, complete definitions of the classes used in the program are provided. The definitions of the classes are provided in the same order as they are described in Chapter 4.

```
_CLASSDEF (Structure)

//Class to store the description of the structure
class Structure
{
public:
  char name[80];      //Name of the structure or job
  int type;           //plane frame = 1, space frame = 2
  int Num_Joint;      //Total number of joints
  int Num_Memb;       //Total number of Members
  int Num_Sprt;       //Total number of Supports
  int Num_HingeSprt;  //Total number of Hinge supports
  int Num_FixedSprt;  //Total number of Fixed supports
  int Num_Load;       //Total number of Load cases
  int Num_LoadComb;   //Total number of load combinations
  int Num_Beam;       //Total number of beams
  int Num_Column;     //Total number of columns
  int Num_BGrp;       //Total number of groups of beam
  float length;       //Length of the structure (in x direction)
  float height;       //Height of the structure (in y direction)
  float depth;        //Width of the structure (in z direction)

  //Transformed maximum and minimum coordinates for graphics
  float x_trans_min, y_trans_min, x_trans_max, y_trans_max;
  int Num_JtLoad;     //Number of joint loads on the structure
  JointLoad *jt_load; //Joint loads' description

  void getdata();       //Gets data from the input file
};


_CLASSDEF (LoadCase)

//Class to store description of the individual load cases
class LoadCase
{
public:
  char type[20];      //string load type
```

```
  //Defining load type
  int load_case; //1 =dl, 2 =ll; 3 =wl, 4 = snow, 5 = earthquake
  int Num_Load;  //Number of load cases
  int Tot_Ptld;  //Number of concentrated loads in the load case
  int Tot_Linear;//Number of linear loads in the load case
  int Tot_Udl;   //Number of uniform loads in the load case
  void Initialize()  //LoadCase()
    {Tot_Ptld = Tot_Linear = Tot_Udl = 0;};
  void getdata1();   //Gets data
};


_CLASSDEF (LoadCombination)

//Class to store data regarding load combinations
class LoadCombination
{
public :
  float *factor;      //Load factor
  int *load_case;     //Load case
  void Initiate();
};


_CLASSDEF (Member)

//Class to store member data
class Member
{
private:
  PtLoad *ptld;              //Applied concentrated loads
  UniLoad *udl;              //Applied uniform loads
  LinearLoad *tri_ld;        //Applied linear loads
  MemberForce *memb_force;  //Member end forces
  int num_tri_ld;  //Total number of linear loads on the member
  int num_ptld;    //Total number of concentrated loads on the
                   //member
  int num_udl;     //Total number of uniform loads on the member
  int number;      //Member number
  int joint_num1;  //Start joint number
  int joint_num2;  //End joint number

  //Actual coordinates of start joint
  float x1, y1, z1;
  //Transformed coordinates of start joint for graphics
  float x1_trans, y1_trans;

  //Actual coordinates of end joint
  float x2, y2, z2;
  //Transformed coordinates of end joint for graphics
  float x2_trans, y2_trans;
```

```
  int dir; //Direction of the member x = 1, y = 2, z = 3;
  float l;              //Length of the member
  float AX;             //Cross sectional area
  float IX, IY, IZ;     //Moment of Inertia
  float E;              //Young's modulus
  float anal_b, anal_h; //Width and depth used in the analysis
  float fc;           //Compressive strength of the concrete
  float fy;           //Yield strength of steel
  int stNum;          //Size of stirrup for beam design
  int tie_size;       //Size of ties for column design
  float cover;        //Cover to the reinforcement

  //Reinforcement option for column design
  int REINF; //1 for 'on four sides', 2 for 'on two sides'

  //Dimension option in beam design
  //1 for 'Use dimen. from analysis', 2 for 'Select new dimen.'
  int OPT_DIM;

  float kbx;      //Eff. length factor for bending about x-axis
  float kby;      //Eff. length factor for bending about y-axis
  float ksx;      //Eff. length factor for sway about x-axis
  float ksy;      //Eff. length factor for sway about y-axis
  float lux;      //Unbraced length about x-axis
  float luy;      //Unbraced length about y-axis

  //Ratio of sum of EI/lc of columns to sum of EI/l of beams
  float sai_xt;  //in x direction, at top of the column
  float sai_xb;  //in x direction, at bottom of the column
  float sai_zt;  //in z direction, at top of the column
  float sai_zb;  //in z direction, at bottom of the column

public:
  friend class Beam;
  friend class Column;

  //Allows the user to change the default values for beams
  friend class TBeamParaDlg;       //For beam design parameters
  //Allows the user to change the default values for columns
  friend class TColumnParaDlg;     //for column design parameters

  int type;  //1 for beam, 2 for column, 0 for other
  int group; //0 for independent member, max. 10 groups allowed
  //Reinf. option for group beam design: 0 for ind. member,
  //1 for only same size, 2 for size and rein. both
  int option;

  void get_joint_coordi();  //Gets joint coordinates
  void get_section();       //Gets section properties
  void get_E();             //Gets modulus of elasticity
  void get_fc_fy();         //Gets fc and fy
  void get_stNum();         //Gets stirrup size
```

```
  void get_cover();          //Gets cover to the reinforcement
  void make_arrays(int memb_num); //Creates arrays dynamically
  void get_tri_ld_data();          //Gets linear load data
  void get_udl_data();             //Gets uniform load data
  void get_ptld_data();            //Gets concentrated load data
  void get_lt_end_forces();        //Gets left end forces
  void get_rt_end_forces();        //Gets right end forces
  float return_length();    //Provides length to other classes
  void cal_k(); //Calculates eff. length factors for columns
  //Solves the equation for k- factors for braced condition
  float BrSolve_k(float sai_t, float sai_b);
  //Solves the equation for k- factors for unbraced condition
  float NBrSolve_k(float sai_t, float sai_b);
  //Calculates transformed end coordinates for graphics
  void transform();
  //Finds range of the transformed coordinates
  void find_trans_range();
  //Provides transformed coordinates of both ends
  void give_trans_coordi_jt_num(float*, float*, float*, float*,
            int*, int*);
};


_CLASSDEF (UniLoad)

//Class to store uniform load data
class UniLoad
{
public:
  int Num_Loading;      //Load case number
  int memb_num;         //Member number on which applied
  char name[40];        //String load type
  int num_udl;          //Number of uniform load
  float udl;            //Intensity of udl
  float udl_a;          //starting point of udl
  float udl_b;          //End point of udl
};


_CLASSDEF (PtLoad)

//Class to store data regarding concentrated load
class PtLoad
{
public:
  int Num_Loading; //Load case number
  int memb_num;    //Member number on which applied
  char name[40];   //String load type
  int num_ptld;    //Number of concentrated load
  float ptld;      //Concentrated load Intensity
  float ptld_a;    //Distance from start of the member
};
```

```
_CLASSDEF (LinearLoad)

//Class to store linear load data
class LinearLoad
{
public:
  char name[40];      //String load type
  int Num_Loading;    //Load case number
  int memb_num;       //Member number on which applied
  int num_tri_ld;     //Number of linear load
  float tri_ld1;      //Intensity of load at start
  float tri_ld2;      //Intensity of load at end
  float tri_ld_a;     //start point of linear load
  float tri_ld_b;     //End point of linear load
};


_CLASSDEF (MemberForce)

//Class to store member end forces
class MemberForce
{
public:
  int Num_Loading;    //Load case number
  char ld_name[40];   //string load type
  int joint_num1;     //start joint number
  int joint_num2;     //End member number
  float af1;          //Axial force at joint 1
  float af2;          //Axial force at joint 2
  float sfy1;         //Shear force at joint 1 in y direction
  float sfy2;         //Shear force at joint 2 in y direction
  float sfz1;         //Shear force at joint 1 in z direction
  float sfz2;         //Shear force at joint 2 in z direction
  float tm1;          //Torsion  moment at joint 1
  float tm2;          //Torsion  moment at joint 2
  float bmy1;         //Bending moment at joint 1  My
  float bmy2;         //Bending moment at joint 2  My
  float bmz1;         //Bending moment at joint 1  Mz
  float bmz2;         //Bending moment at joint 2  Mz
};


_CLASSDEF (Joint)

class Joint
{
private:
  int number;                          //Joint number
  float coordi_x, coordi_y, coordi_z;  //Joint coordinates
  Displacement *displacement;          //Joint displacements
```

```
  JointLoad *jt_load;                    //Joint loads
public:
  friend class Column;
  friend class Beam;
  friend class Member;
  int type;              //type = 1 if support else 0
  void get_type();       //Gets type of the joint
  void getdata();        //Gets data of joint coordinates etc.
  void make_array();         //Creates arrays dynamically
  void get_displacements(); //Gets displacement data
};


_CLASSDEF (Displacement)

//Class to store joint displacements
class Displacement
{
public:
  int Num_Loading;       //Load case number
  char ld_name[40];      //string load type
  //displacements in x, y and z directions
  float disp_x, disp_y, disp_z;
  //Rotations in x, y and z directions
  float rotat_x, rotat_y, rotat_z;
};


_CLASSDEF (JointLoad)

//Class to store joint load data
class JointLoad
{
public:
  int Num_Loading;
  int jt_num;
  int dir;
  float jt_ld;
};


_CLASSDEF (Beam);

//Class for the beam design
class Beam
{
private:
  int memb_num;     //Member number
  int grp_num;      //Group number, 0 for ind. member
  //Reinf. option for group beam design: 0 for ind. member,
  //1 for only same size, 2 for size and rein. both
  int option;
```

```
  float l, E, I; //Length, Young, modulus and moment of inertia

  float num_ptld;     //Total number of point loads on the beam
  float num_tri_ld;   //Total number of linear loads on the beam
  float num_udl;      //Total number of uniform loads on beam
  PtLoad *ptld;       //Concentrated Load data
  LinearLoad *tri_ld; //Linear load data
  UniLoad *udl;       //Uniform load data
  float *ax_force;    //Axial force
  float *lt_mom;      //Left end moment
  float *rt_mom;      //Right end moment

  float *sfx;       //SF at section x for each load
  float *bmx;       //BM at section x for each load
  float *dflx;      //Deflection at section x for each load
  float *tot_sfx;   //Total SF at section x for each load case
  float *tot_bmx;   //Total BM at section x for each load case
  float *tot_dflx; //Total defl. at sect. x for each load case

  float *design_sfx;    //Design SF at section x
  float *design_pbmx;   //Design positive BM at section x
  float *design_nbmx;   //Design positive BM at section x
  float *design_dflx;   //Deflection at section x
  float Nu;             //Axial force

  //Renaming design_pbmx, design_nbmx and design_sfx
  float PMx[MAX_SECT], NMx[MAX_SECT], maxV[MAX_SECT];

  //Steel ratios design, balance, max. and min. respectively
  float rho, rhobal, rhomax, rhomin;
  float phi;        //strength reduction factor
  float fc;         //Compressive strength of concrete
  float fy;         //Yield strength of steel
  float beta1;
  float anal_b, anal_h; //width and total depth used in analysis
  float b, d;           //Width and effective depth
  float cover;          //Cover to the reinforcement

  int tot_try;      //Total number set of trial width and depth
  float trial_h[15]; //trial depth
  float trial_b[15]; //trial width

  boolean chksingly; //'true' if singly reinforced else 'false'
  boolean chkshear;  //'true' if shear check OK else 'false'
  int stNum;                 //Stirrup size
  float Sv[MAX_SECT];        //Stirrup spacing
  float Mu;                  //Maximum bending moment on beam
  float Astx[MAX_SECT];      //Area of steel at top
  float Ascx[MAX_SECT];      //Area of steel at bottom

public:
  Beam(){};    //Constructors for independent beam case
```

```
Beam(int);   //Constructor for beam in a group
friend class TSectionDlg;
friend class TInfDblyDlg;
friend class BeamOutput;    //For saving design results
TSectionDlg *section;       //section selection dialog
//Dialog to ask user to except doubly reinf. section or not
TInfDblyDlg *InfDblyDlg;

//1 for 'dimensions from analysis', 2 for 'new dimensions'
int OPT_DIM;

boolean IsDoubly;//'true' if doubly is OK else 'false'

//Definitions of the functions
//Gets data from the corresponding member object
void get_memb_data(int memb_num);
//Stores common properties in the case of group
void get_common_prop(int);
//Finds moments and shear forces at sections
void analyze();
//Creates arrays dynamically
void make_arrays();
//Calculates moments and shear forces due to left end mom.
void cal_ltmom();
//Calculates moments and shear forces due to right end mom.
void cal_rtmom();
//Calculates moments and shear forces due to uniform load
void cal_udl();
//Calculates moments and shear forces due to concentrated load
void cal_ptld();
//Calculates moments and shear forces due to linear load
void cal_tri_ld();
//Superposes moments and shear forces
void impose(int num_ld);
//Finds max. shear forces and bending moments at sections
void design_forces();
//Renames design forces
void rename_design_forces();
//Deletes arrays and frees memory
void delete_arrays();
//Calculates beta1 according to ACI 318-89
void Beta1();
//Finds maximum absolute moment
void Maximum();
//Finds maximum absolute moment for the group
void absMaxBm(int ct);
//Finds maximum absolute axial force
void MaxNu();
//Finds steel ratios
void steel_ratios();
//Finds trial depth and width
void Findbd();
```

```
  //Returns zero if selected width is zero
  boolean IsWidthZero();
  //Obtains data like b, d, rho etc. from beam[0] in groups
  void get_data();
  //Checks whether section is singly or not
  boolean IsSingly();
  //Finds area of tensile steel for singly rein. beam
  void FindAstx();
  //Finds area of tens. and comp. steel for doubly rein. beam
  void FindAsx();
  //Finds spacing of stirrups
  //Returns 'true' if shear check is satisfied
  boolean shearDesign();
  //Finds max. steel for group if 'Same size and Reinf.' is
  //selected
  void find_max_steel(int group); //stores in beam[0]
  void get_max_steel(); //Gets area of steel from beam[0]
};


_CLASSDEF (BeamOutput)

//Class to store output of beam design
class BeamOutput
{
public:
  int memb_num;  //Member Number
  int group;     //Group Number
  int option; //1 = 'Same size only', 2 = 'Same size and Reinf.'
  float l;       //Length
  float b, d, h; //Section dimensions
  float fc, fy;  //Material strength
  int stNum;     //Stirrup size
  //steel area required at top and bottom
  //and spacing of stirrup at all sections
  float Ascx[MAX_SECT], Astx[MAX_SECT], Sv[MAX_SECT];
  //positive and negative design BM and SF at all sections
  float PMx[MAX_SECT], NMx[MAX_SECT], maxV[MAX_SECT];

  //Gets data from beam object in independent beam case
  void get_data();
  //Gets data from beam object for Group case
  void get_data(int beam_num);
};


_CLASSDEF (Column)

//Class to design columns
class Column
{
private:
```

```
int memb_num;   //Member number
int REINF;      //1 = on four sides, 2 = on two sides
float l;        //Length of column
float E;        //Modulus of elasticity
float I;        //Moment of inertia
float h;        //Dim. of column section parallel to y-axis
float b;        //Dim. of column section parallel to x-axis
float Ag;       //Gross area of section (AX of member object)
float cover;    //Cover to the centroid of the bars

float kbx;      //Eff. length factor for bending about x-axis
float kby;      //Eff. length factor for bending about y-axis
float ksx;      //Eff. length factor for sway about x-axis
float ksy;      //Eff. length factor for sway about y-axis
float lux;      //Unbraced length about x-axis
float luy;      //Unbraced length about y-axis

int joint1, joint2; //Start and end joint numbers

float fy ;          //Yield strength of steel
float fc;           //Compressive strength of concrete
float beta1;        //As per ACI 318-89 specifications
float phi;          //strength reduction factor

float *Pd;          //Factored dead load
float *Pu;          //Factored axial load
float *betaD;       //As per ACI 318-89 specifications
float *sigma_Pu, *sigma_Pcx, *sigma_Pcy;
float *Pu1;  //Allowable axial load
float *Pn;   //Nominal design axial load
float *M1bx; //Smaller factored braced end moment about x-axis
float *M2bx; //Larger factored end braced moment about x-axis
float *M1by; //Smaller factored end braced moment about y-axis
float *M2by; //Larger factored end braced moment about y-axis
float *M1sx; //Smaller factored end sway moment about x-axis
float *M2sx; //Larger factored end sway moment about x-axis
float *M1sy; //Smaller factored end sway moment about y-axis
float *M2sy; //Larger factored end sway moment about y-axis
float *Mn;   //Resultant applied moment
float *Mn1;  //Allowable moment

//Moment Magnification factors for each load combination
float *DeltaBx, *DeltaBy, *DeltaSx, *DeltaSy;

float *Rhog; //Final steel ratio for each load combination
float *Ast;  //Area of steel required for each load comb.

int critical_comb;    //Critical load combination for design
//Area of steel and steel ratio for the critical load comb.
float final_Ast, final_Rhog;
//Critical applied axial load and allowable axial load
float final_Pu, final_Pu1;
```

```
  //Critical moments
  float final_M1bx, final_M2bx, final_M1by, final_M2by;
  float final_M1sx, final_M2sx, final_M1sy, final_M2sy;
  float final_Mn, final_Mn1;
  //Critical moment magnification factors
  float final_deltaBx, final_deltaBy; //for bending
  float final_deltaSx, final_deltaSy; //for sway

  float x[21];   //x-coordinates of the bars
  float y[21];   //y-coordinates of the bars

  Steel *steel, final_steel;  //Bars sizes, numbers and area
  int tie_size;           //Size of tie
  float tie_spacing;      //Spacing of ties
  int isDesignOk;         //1 if steel% is less than 8% else 0

public:
  //Constructor
  Column(int memb_num);   //Gets data from member object
  ~Column(); //Destructor, frees memory occupied by itself
  //Provides data to ColumnOutput object
  friend class ColumnOutput;
  //Creates arrays dynamically
  void make_arrays();
  //Frees memory dynamically allocated for arrays
  void delete_arrays();
  void design()     //Designs column
  void Beta1();     //Calculates beta1 according to ACI 318-89
  //Places bars for creating interaction diagrams
  void BarPlacement();
  //Finds longitudinal steel for each load combination
  void find_steel(int num_ld);
  //Magnifies bending moment in case of a long column
  void Magnify_b(float *mu, float *deltaB, float beta_d, float
    pu, float m1b, float size, float k, float lu, float rhog);
  //Magnifies sway moment in case of a long column
  void Magnify_s(float *mu, float *deltaB, float sigma_Pu,
                 float sigam_Pc);
  //Checks suitability by the reciprocal method
  void Reciprocal(float pu, float mu, float dim1, float dim2,
            float dist[], float rhog, float *pu1, float *pu2);
  //Picks critical values from all load combinations
  void pick_critical();

  //Steel Routines
  //Finds bar combination
  void find_bars(float ast_face, float least_dim);
  //Checks whether bars fits into the section or not
  int check_width(float least_dim, int size1, int num1,
                  int size2, int num2);
  //Checks possible bar combination is symmetric
  int check_symmetry(int n1, int n2);
```

```
  //Sorts bar combination in ascending order of area of steel
  void sort(int tot_comb);
  //Picks one bar size if within 5% of required steel
  //else picks bar combination with the least steel
  void pick_most_ecconomical(int tot_comb, float ast_face);
  //Designs lateral reinforcement, ties
  void design_ties();
};


_CLASSDEF (ColumnOutput)

//Class to store output of column design
class ColumnOutput
{
public:
  int memb_num;     //Member number
  int REINF;        //1 = on four sides, 2 = on two sides
  float l;          //Length of the column
  float b, h;       //Section dimensions
  float fc, fy;     //Material strength

  int isDesignOk;     //Flag for steel% <= 8.0% or not
  int critical_comb;  //Critical load combination for design
  float Pu;           //factored axial load
  float Pu1;          //allowable axial load
  float Pn;
  float M1bx;  //smaller factored braced end moment about x-axis
  float M2bx;  //larger factored end braced moment about x-axis
  float M1by;  //smaller factored end braced moment about y-axis
  float M2by;  //larger factored end braced moment about y-axis

  float M1sx;  //smaller factored sway end moment about x-axis
  float M2sx;  //larger factored end sway moment about x-axis
  float M1sy;  //smaller factored end sway moment about y-axis
  float M2sy;  //larger factored end sway moment about y-axis

  float Mn;    //resultant applied moment
  float Mn1;   //allowable moment

  //Effective length factors
  float kbx, kby, ksx, ksy;
  //Moment magnification factors
  float deltaBx, deltaBy, deltaSx, deltaSy;

  float Rhog;           //Final steel ratios
  float Ast;            //Area of steel required

  Steel final_steel;    //Final bar combination selected
  int tie_size;         //Size of ties
  float tie_spacing;    //Spacing of ties
```

```
  void get_data();      //Gets data from the column object
};


_CLASSDEF (TPostproAPP);

//Class for post processor application
class TPostproApp : public TApplication
{
public:
  //Constructor
  TPostproApp(LPSTR AName, HINSTANCE hInstance, HINSTANCE
    hPrevInstance, LPSTR lpCmdLine, int nCmdShow)
      : TApplication(AName, hInstance, hPrevInstance,
          lpCmdLine, nCmdShow) {};
  //Initializes main window of the application
  virtual void InitMainWindow();
  virtual void InitInstance();
  //Initializes structure class when application is started
  virtual void InitStructure();
  //Initializes group data when application starts
  virtual void InitGroup();
};


_CLASSDEF(TPostproWindow);

//Class to define main window behavior
class TPostproWindow : public TWindow
{
private:
  //Input and output file names
  char infilename[MAXPATH], outfilename[MAXPATH];
  int fileSaved, printExecuted; //Save and print flags
  int beam_disp, column_disp;   //Display flags
  int define_loadcases;         //Load case flag
public:
  //Constructor
  TPostproWindow(PTWindowsObject AParent, LPSTR ATitle);

  //Sets attributes of applications window
  void setAttr();
  //Paints the main window of the application
  virtual void Paint(HDC PaintDC, PAINTSTRUCT _FAR & PaintInfo);
  virtual void GetWindowClass(WNDCLASS _FAR & wndClass);

  //When 'File/Open' menu is selected
  void FileOpen(RTMessage Msg) = [CM_FIRST + CM_FileOpen];
  //When 'File/Save' menu is selected
  void FileSave(RTMessage Msg) = [CM_FIRST + CM_FileSave];
  //When 'File/Save As' menu is selected
  void FileSaveAs(RTMessage Msg) = [CM_FIRST + CM_FileSaveAs];
```

```
  //When 'File/Print' menu is selected
  void FilePrint(RTMessage Msg) = [CM_FIRST + CM_FilePrint];
  //When File/Exit' menu is selected
  void FileExit(RTMessage Msg) = [CM_FIRST + CM_FileExit];

  //When 'Load Cases/Define Load Cases' menu is selected
  void DefineLoadCases(RTMessage Msg) = [CM_FIRST +
        CM_DefineLoadCases];

  //When 'Beams/Design Parameters' menu is selected
  void BeamPara(RTMessage Msg) = [CM_FIRST + CM_BeamPara];
  //When 'Beams/Mark Group' menu is selected
  void BeamMarkGroup(RTMessage Msg) = [CM_FIRST +
        CM_BeamMarkGroup];
  //When 'Columns/Design Parameters' menu is selected
  void ColumnPara(RTMessage Msg) = [CM_FIRST + CM_ColumnPara];

  //When 'Design/Beams' menu is selected
  void DesignBeams(RTMessage Msg) = [CM_FIRST + CM_DesignBeams];
  //When 'Design/Columns' menu is selected
  void DesignColumns(RTMessage Msg) = [CM_FIRST +
        CM_DesignColumns];

  //When 'Results/Beams' menu is selected
  void ViewBeamOutput(RTMessage Msg) = [CM_FIRST + CM_ViewBeam];
  //When 'Results/Columns' menu is selected
  void ViewColumnOutput(RTMessage Msg) = [CM_FIRST +
        CM_ViewColumn];

  //When 'Help/About' Menu is selected
  void HelpAbout(RTMessage Msg) = [CM_FIRST + CM_HelpAbout];

  void save_file();  //Saves output file
};


_CLASSDEF (TBeamResultsChild)

//Class to define beam results child window behavior
class TBeamResultsChild : public TWindow
{
public:
  //Constructor
  TBeamResultsChild(PTWindowsObject AParent, LPSTR ATitle);
  //Paints the child window
  virtual void Paint(HDC PaintDC, PAINTSTRUCT _FAR & PaintInfo);
  //Sets attributes of the child window
  void setAttr();
  virtual void GetWindowClass(WNDCLASS _FAR & wndClass);

  //When 'Next' menu is selected
  void NextPressed(RTMessage Msg) = [CM_FIRST + CM_Next];
```

```
  //When 'Prev' menu is selected
  void PrevPressed(RTMessage Msg) = [CM_FIRST + CM_Prev];
  //When 'GoTo' menu is selected
  void GoToPressed(RTMessage Msg) = [CM_FIRST + CM_GoTo];
  //When 'Done' menu is selected
  void DonePressed(RTMessage Msg) = [CM_FIRST + CM_Done];
};


_CLASSDEF (TColumnResultsChild)

//Class to define column results child window behavior
class TColumnResultsChild : public TWindow
{
public:
  //Constructor
  TColumnResultsChild(PTWindowsObject AParent, LPSTR ATitle);
  //Paints the child window
  virtual void Paint(HDC PaintDC, PAINTSTRUCT _FAR & PaintInfo);
  //Sets attributes of the child window
  void setAttr();
  virtual void GetWindowClass(WNDCLASS _FAR & wndClass);
  //When 'Next' menu is selected
  void NextPressed(RTMessage Msg) = [CM_FIRST + CM_Next];
  //When 'Prev' menu is selected
  void PrevPressed(RTMessage Msg) = [CM_FIRST + CM_Prev];
  //When 'GoTo' menu is selected
  void GoToPressed(RTMessage Msg) = [CM_FIRST + CM_GoTo];
  //When 'Done' menu is selected
  void DonePressed(RTMessage Msg) = [CM_FIRST + CM_Done];
};


_CLASSDEF (TColumnDesignChild)

//Class to define column design progress child window behavior
class TColumnDesignChild : public TWindow
{
public:
  //Constructor
  TColumnDesignChild(PTWindowsObject AParent, LPSTR ATitle);
  //Paints the child window
  virtual void Paint(HDC PaintDC, PAINTSTRUCT _FAR & PaintInfo);
  //Sets attributes of the child window
  void setAttr();
};


_CLASSDEF (TFileDlg);

//class definition for the file dialog
class TFileDlg : public TFileDialog
```

```
{
public:
  TFileDlg (PTWindowsObject AParent, int ResourceId, LPSTR
    AFilePath) : TFileDialog(AParent, ResourceId, AFilePath){};
};


_CLASSDEF (THelpDlg)

//Help About Dialog class
class THelpDlg : public TDialog
{
public:
  //Constructor
  THelpDlg(PTWindowsObject AParent, LPSTR AName)
    : TDialog(AParent, AName){};
};


_CLASSDEF (TLoadCaseDlg)

// Load case dialog class
class TLoadCaseDlg : public TDialog
{
private :
  TStatic *Static1;     //Static window
public :
  TLoadCaseDlg(PTWindowsObject AParent, LPSTR ATitle);
  //Initializes dialog box
  virtual void WMInitDialog(RTMessage Msg) = [WM_FIRST +
      WM_INITDIALOG];
  //When 'OK' button is pressed
  void OK(RTMessage Msg) = [ID_FIRST + IDOK];

//Radio button to select dead load case
  virtual void Radio1(RTMessage Msg) = [ID_FIRST + ID_Radio1];
//Radio button to select live load case
  virtual void Radio2(RTMessage Msg) = [ID_FIRST + ID_Radio2];
//Radio button to select wind load case
  virtual void Radio3(RTMessage Msg) = [ID_FIRST + ID_Radio3];
//Radio button to select snow load case
  virtual void Radio4(RTMessage Msg) = [ID_FIRST + ID_Radio4];
//Radio button to select earthquake load case
  virtual void Radio5(RTMessage Msg) = [ID_FIRST + ID_Radio5];
};


_CLASSDEF (TBeamMarkGroupDlg)

//Class for beam mark group dialog
class TBeamMarkGroupDlg : public TDialog
{
```

```
private :
  int count;
  int first[11];
  char changeList[80];
  //Stores member numbers for each load cases
  char buff[11][80];
  TStatic *Static1, *Static2; //Static windows
  TComboBox *numberList;    //Combo box for beam member numbers
public :
  //Constructor
  TBeamMarkGroupDlg(PTWindowsObject AParent, LPSTR ATitle);
  //Initializes dialog
  virtual void WMInitDialog(RTMessage Msg) = [WM_FIRST +
        WM_INITDIALOG];
  //Sets group data when dialog is displayed or 'Next' or
  //'Prev' is pressed
  void SetGroup(int count);

  //Radio button to select 'Same size only' option
  virtual void Radio1(RTMessage Msg) = [ID_FIRST + ID_Radio1];
  //Radio button to select 'Same size and reinf.' option
  virtual void Radio2(RTMessage Msg) = [ID_FIRST + ID_Radio2];

  //Adds selected member into the group member list
  void PlusPressed(RTMessage Msg) = [ID_FIRST + ID_Plus];
  /Adds member when left button of the mouse is double clicked
  void LDblClked(RTMessage) = [WM_FIRST + WM_LBUTTONDOWN];

  //Deletes selected member from the group member list
  void MinusPressed(RTMessage Msg) = [ID_FIRST + ID_Minus];
  //Deletes member when right button is double clicked
  void RDblClked(RTMessage) = [WM_FIRST + WM_RBUTTONDBLCLK];

  //Goes to next group when 'Next' is pressed
  void NextPressed(RTMessage Msg) = [ID_FIRST + ID_Next];
  //Goes to previous group when 'Prev' is pressed
  void PrevPressed(RTMessage Msg) = [ID_FIRST + ID_Prev];
  //Closes dialog and store the data, when 'Cancel' is pressed
  void DonePressed(RTMessage Msg) = [ID_FIRST + ID_Cancel];
};


_CLASSDEF (TBeamParaDlg)

//Class for Beam Design Parameters Dialog
class TBeamParaDlg : public TDialog
{
private:
  TEdit *Edit1, *Edit2, *Edit3, *Edit4;  //Edit windows
  TComboBox *numberList;  //Combo Box for beam member numbers

public:
```

```
  //Constructor
  TBeamParaDlg(PTWindowsObject AParent, LPSTR ATitle);
  //Initializes the dialog box
  virtual void WMInitDialog(RTMessage Msg) = [WM_FIRST +
          WM_INITDIALOG];
  //When 'OK' is pressed
  virtual void OK(RTMessage Msg) = [ID_FIRST + IDOK];

  //Radio button for 'Use dimensions from the analysis' option
  virtual void Radio1(RTMessage Msg) = [ID_FIRST + ID_Radio1];
  //Radio button for 'Select new dimensions' option
  virtual void Radio2(RTMessage Msg) = [ID_FIRST + ID_Radio2];
//Handles the selection from the list of member numbers
  virtual void HandleListBoxMsg(RTMessage Msg) = [ID_FIRST +
          ID_Combo];
};


_CLASSDEF (TColumnParaDlg)

//Class for column design parameters dialog box
class TColumnParaDlg : public TDialog
{
private :
  TEdit *Edit1, *Edit2, *Edit3;    //Edit Windows
  TComboBox *numberList;  //Combo Box for column member numbers

public :
  //Constructor
  TColumnParaDlg(PTWindowsObject AParent, LPSTR ATitle);
  //Initializes dialog
  virtual void WMInitDialog(RTMessage Msg) = [WM_FIRST +
        WM_INITDIALOG];
  //When 'OK' is pressed
  virtual void OK(RTMessage Msg) = [ID_FIRST + IDOK];

  //Radio button to select 'Reinforcement on two sides' option
  virtual void Radio1(RTMessage Msg) = [ID_FIRST + ID_Radio1];
  //Radio button to select 'Reinforcement on four sides' option
  virtual void Radio2(RTMessage Msg) = [ID_FIRST + ID_Radio2];
  //Handles the selection from the list of member numbers
  virtual void HandleListBoxMsg(RTMessage Msg) = [ID_FIRST +
        ID_Combo];
};


_CLASSDEF(TSectionDlg);

//Section selection dialog box
class TSectionDlg : public TDialog
{
private:
```

```
  TEdit *Edit1, *Edit2;     //Edit windows
  TStatic *Static1, *Static2, *Static3;  //Static Windows
  TListBox *sectList;       //List box
  int memb_num;             //Member number
  int grp_num;              //Group number
  float cover;              //Cover to the reinforcement
  int tot_try;      //Total number set of trial width and depth
  float trial_h[15];        //trial depth
  float trial_b[15];        //trial width
  float b, d;               //width and depth of the beam
  float anal_b, anal_h; //width and depth used in the analysis
public :
  //Constructor
  TSectionDlg(PTWindowsObject AParent, LPSTR ATitle);
  //Initializes the dialog box
  virtual void WMInitDialog(RTMessage Msg) = [WM_FIRST +
      WM_INITDIALOG];
  void get_data(); //Gets data
  void give_bd();  //Provides b and d to other object
  //When 'OK' is pressed
  void OK(RTMessage Msg) = [ID_FIRST + IDOK];
  //When 'Cancel' is pressed
  void Cancel(RTMessage Msg) = [ID_FIRST + IDCANCEL];
  //Handles the list box selection
  virtual void HandleListBoxMsg(RTMessage Msg) = [ID_FIRST +
      ID_List1];
};


_CLASSDEF (TInfDblyDlg)
//Doubly Reinforced section Inform Dialog Class
class TInfDblyDlg : public TDialog
{
public :
  char IsDoubly;
  //Constructor
  TInfDblyDlg :: TInfDblyDlg(PTWindowsObject AParent, LPSTR
                        AName) : TDialog(AParent, AName){};

  //When 'OK' is pressed
  void doublyOK(RTMessage Msg) = [ID_FIRST + IDYES];
  //When 'Cancel' is pressed
  void closeDoubly(RTMessage Msg) = [ID_FIRST + IDNO];
};


_CLASSDEF (TBeamMembNumDlg)

//Beam member number dialog class
class TBeamMembNumDlg : public TDialog
{
private:
```

```
  TComboBox *numberList;    //Combo box to list member numbers
public:
  //Constructor
  TBeamMembNumDlg(PTWindowsObject AParent, LPSTR ATitle);
  //Initializes the dialog box
  virtual void WMInitDialog(RTMessage Msg) = [WM_FIRST +
      WM_INITDIALOG];
  //When 'OK' button is pressed
  void OK(RTMessage Msg) = [ID_FIRST + IDOK];
  //When 'Cancel' button is pressed
  void Cancel(RTMessage Msg) = [ID_FIRST + IDCANCEL];
};


_CLASSDEF (TColumnNumDlg)

// Column member number dialog class
class TColumnMembNumDlg : public TDialog
{
private:
  TComboBox *numberList;    //Combo box to list member numbers
public:
  //Constructor
  TColumnMembNumDlg(PTWindowsObject AParent, LPSTR ATitle);
  //Initializes the dialog box
  virtual void WMInitDialog(RTMessage Msg) = [WM_FIRST +
      WM_INITDIALOG];
  //When 'OK' button is pressed
  void OK(RTMessage Msg) = [ID_FIRST + IDOK];
  //When 'Cancel' button is pressed
  void Cancel(RTMessage Msg) = [ID_FIRST + IDCANCEL];
};
```

Appendix B
Input and Output Files for Test Problems

In this Appendix, the input and output files for the two test problems discussed in Chapter 5 are provided. The first test structure was a continuous beam while the second test structure was a two story space frame. The input and output files for the continuous beam problem are given first followed by those for the two story space frame.

B.1 Input and Output Files for Continuous Beam

B.1.1 Input File for Continuous Beam

```
STRUCTURE Continuous Beam - Test Problem 1
TYPE PLANE FRAME
METHOD STIFFNESS
TABULATE ALL
NUMBER OF JOINTS     5
NUMBER OF MEMBERS    4
NUMBER OF SUPPORTS   5
NUMBER OF LOADINGS   2
NUMBER OF LOAD COMBINATIONS   1
JOINT COORDINATES
 1     0.00   0.00 S
 2   282.00   0.00 S
 3   582.00   0.00 S
 4   882.00   0.00 S
 5  1164.00   0.00 S
MEMBER INCIDENCES
 1   1   2
 2   2   3
 3   3   4
 4   4   5
MEMBER PROPERTIES PRISMATIC
 1 AX IZ 468.00 26364.00
 2 AX IZ 468.00 26364.00
 3 AX IZ 468.00 26364.00
 4 AX IZ 468.00 26364.00
CONSTANTS E 3605.00 ALL
JOINT RELEASES
 1 MOMENT Z
 2 MOMENT Z
 3 MOMENT Z
 4 MOMENT Z
 5 MOMENT Z
LOADING 1 - dead load
MEMBER LOADS
```

```
 1 FORCE Y UNIFORM -0.207
 2 FORCE Y UNIFORM -0.207
 3 FORCE Y UNIFORM -0.207
 4 FORCE Y UNIFORM -0.207
LOADING 2 - live load
MEMBER LOADS
 1 FORCE Y UNIFORM -0.250
 2 FORCE Y UNIFORM -0.250
 3 FORCE Y UNIFORM -0.250
 4 FORCE Y UNIFORM -0.250
 1 FORCE Y CONCENTRATED -20.00 141.00
 2 FORCE Y CONCENTRATED -20.00 150.00
 3 FORCE Y CONCENTRATED -20.00 150.00
 4 FORCE Y CONCENTRATED -20.00 141.00
LOAD COMBINATION 1 - 1.4 * 1 + 1.7 * 2
MEMBER LOADS
 1 FORCE Y UNIFORM -0.715
 2 FORCE Y UNIFORM -0.715
 3 FORCE Y UNIFORM -0.715
 4 FORCE Y UNIFORM -0.715
 1 FORCE Y CONCENTRATED -34.00 141.00
 2 FORCE Y CONCENTRATED -34.00 150.00
 3 FORCE Y CONCENTRATED -34.00 150.00
 4 FORCE Y CONCENTRATED -34.00 141.00
SOLVE

STRUCTURE Continuous Beam - Test Problem 1
=================================================

LOADING 1 - dead load
=================================================

MEMBER FORCES

MEMBER JOINT AXIAL FORCE SHEAR FORCE MOMENT
 1       1     0.00         22.71       0.00
 1       2     0.00         35.74   -1836.14
 2       2     0.00         32.50    1836.14
 2       3     0.00         29.69   -1414.05
 3       3     0.00         30.07    1414.05
 3       4     0.00         32.93   -1860.06
 4       4     0.00         36.21    1860.06
 4       5     0.00         23.01       0.00

        SUPPORT JOINT DISPLACEMENTS

JOINT X-DISPLACEMENT Y-DISPLACEMENT ROTATION
 1    0.0000E+00     0.0000E+00    -0.7950E-05
 2    0.0000E+00     0.0000E+00     0.1562E-05
 3    0.0000E+00     0.0000E+00     0.8704E-20
 4    0.0000E+00     0.0000E+00    -0.1562E-05
 5    0.0000E+00     0.0000E+00     0.7950E-05

STRUCTURE Continuous Beam - Test Problem 1
=================================================
```

```
LOADING 2 - live load
==================================================

MEMBER FORCES

MEMBER JOINT AXIAL FORCE SHEAR FORCE MOMENT
 1       1      0.00          34.13        0.00
 1       2      0.00          56.37    -3135.39
 2       2      0.00          50.05     3135.39
 2       3      0.00          44.95    -2369.80
 3       3      0.00          44.95     2369.80
 3       4      0.00          50.05    -3135.39
 4       4      0.00          56.37     3135.39
 4       5      0.00          34.13        0.00

        SUPPORT JOINT DISPLACEMENTS

JOINT X-DISPLACEMENT Y-DISPLACEMENT ROTATION
 1     0.0000E+00      0.0000E+00  -0.1356E-04
 2     0.0000E+00      0.0000E+00   0.2797E-05
 3     0.0000E+00      0.0000E+00   0.1138E-19
 4     0.0000E+00      0.0000E+00  -0.2797E-05
 5     0.0000E+00      0.0000E+00   0.1356E-04

STRUCTURE Continuous Beam - Test Problem 1
==================================================

LOAD COMBINATION 1 - 1.4 * 1 + 1.7 * 2
==================================================

MEMBER FORCES

MEMBER JOINT AXIAL FORCE SHEAR FORCE MOMENT
 1       1       0.00         89.83        0.00
 1       2       0.00        145.86    -7900.60
 2       2       0.00        130.59     7900.60
 2       3       0.00        117.97    -6008.20
 3       3       0.00        117.97     6008.20
 3       4       0.00        130.59    -7900.60
 4       4       0.00        145.86     7900.60
 4       5       0.00         89.83        0.00

        SUPPORT JOINT DISPLACEMENTS

JOINT X-DISPLACEMENT Y-DISPLACEMENT ROTATION
 1    0.0000E+00      0.0000E+00    -0.3403E-04
 2    0.0000E+00      0.0000E+00     0.6888E-05
 3    0.0000E+00      0.0000E+00     0.8798E-07
 4    0.0000E+00      0.0000E+00    -0.7240E-05
 5    0.0000E+00      0.0000E+00     0.3454E-04
```

B.1.2 Output File for the Continuous Beam

Design results for structure: Continuous beam - test problem 1

Beam design results

Member # 1 type: Beam Independent member

width = 18.0 inches f'c = 4.0 ksi Stirrup size = # 4
Depth = 26.0 inches fy = 60.0 ksi

Distance From Left End (ft)	Design Moment Negative (kip-ft)	Design Moment Positive (kip-ft)	Design Shear Force (kip)	Area of steel Bottom (sq.in.)	Area of steel Top (sq.in.)	Stirrup Spacing (inch)
0.0	-	-	73.0	-	-	11.8
1.7	-	138.6	73.0	-	1.41	11.8
3.4	-	253.1	61.0	-	2.53	11.8
5.0	-	343.4	46.6	-	3.50	11.8
6.7	-	409.5	32.2	-	4.25	11.8
8.4	-	451.5	17.8	-	4.74	-
10.1	-	469.3	3.4	-	4.95	-
11.8	-	462.8	-45.0	-	4.87	11.8
13.4	-	375.2	-59.4	-	3.86	11.8
15.1	-	263.4	-73.8	-	2.64	11.8
16.8	-	127.4	-88.2	-	1.41	11.0
18.5	32.8	-	-102.6	1.41	-	8.2
20.1	217.2	-	-117.0	2.15	-	6.6
21.8	425.7	-	-129.0	4.44	-	5.6
23.5	658.4	-	-129.0	7.55	-	5.6

Member # 2 type: Beam Independent member

width = 18.0 inches f'c = 4.0 ksi Stirrup size = # 4
Depth = 26.0 inches fy = 60.0 ksi

Distance From Left End (ft)	Design Moment Negative (kip-ft)	Design Moment Positive (kip-ft)	Design Shear Force (kip)	Area of steel Bottom (sq.in.)	Area of steel Top (sq.in.)	Stirrup Spacing (inch)
0.0	658.4	-	113.8	7.55	-	6.9
1.8	438.9	-	113.8	4.59	-	6.9
3.6	246.8	-	99.9	2.46	-	8.6
5.4	82.1	-	84.6	1.41	-	11.8
7.1	-	55.3	69.3	-	1.41	11.8
8.9	-	165.3	54.0	-	1.62	11.8
10.7	-	248.0	38.6	-	2.47	11.8
12.5	-	303.3	-10.7	-	3.06	-
14.3	-	270.5	-26.0	-	2.71	11.8
16.1	-	210.4	-41.3	-	2.08	11.8
17.9	-	122.9	-56.7	-	1.41	11.8
19.6	-	8.0	-72.0	-	1.41	11.8
21.4	134.2	-	-87.3	1.41	-	11.3
23.2	303.8	-	-101.1	3.07	-	8.5
25.0	500.7	-	-101.1	5.46	-	8.5

Output File (Contd.)

Design results for structure: Continuous beam - test problem 1

Beam design results

Member # 3 type: Beam Independent member

width = 18.0 inches f'c = 4.0 ksi Stirrup size = # 4
Depth = 26.0 inches fy = 60.0 ksi

Distance From Left End (ft)	Design Moment Negative (kip-ft)	Design Moment Positive (kip-ft)	Design Shear Force (kip)	Area of steel Bottom (sq.in.)	Area of steel Top (sq.in.)	Stirrup Spacing (inch)
0.0	500.7	-	101.1	5.46	-	8.5
1.8	303.8	-	101.1	3.07	-	8.5
3.6	134.2	-	87.3	1.41	-	11.3
5.4	-	8.0	72.0	-	1.41	11.8
7.1	-	122.9	56.7	-	1.41	11.8
8.9	-	210.4	41.3	-	2.08	11.8
10.7	-	270.5	26.0	-	2.71	11.8
12.5	-	303.3	-23.3	-	3.06	11.8
14.3	-	248.0	-38.6	-	2.47	11.8
16.1	-	165.3	-54.0	-	1.62	11.8
17.9	-	55.3	-69.3	-	1.41	11.8
19.6	82.1	-	-84.6	1.41	-	11.8
21.4	246.8	-	-99.9	2.46	-	8.6
23.2	438.9	-	-113.8	4.59	-	6.9
25.0	658.4	-	-113.8	7.55	-	6.9

Member # 4 type: Beam Independent member

width = 18.0 inches f'c = 4.0 ksi Stirrup size = # 4
Depth = 26.0 inches fy = 60.0 ksi

Distance From Left End (ft)	Design Moment Negative (kip-ft)	Design Moment Positive (kip-ft)	Design Shear Force (kip)	Area of steel Bottom (sq.in.)	Area of steel Top (sq.in.)	Stirrup Spacing (inch)
0.0	658.4	-	129.0	7.55	-	5.6
1.7	425.7	-	129.0	4.44	-	5.6
3.4	217.2	-	117.0	2.15	-	6.6
5.0	32.8	-	102.6	1.41	-	8.2
6.7	-	127.4	88.2	-	1.41	11.0
8.4	-	263.4	73.8	-	2.64	11.8
10.1	-	375.2	59.4	-	3.86	11.8
11.8	-	462.8	11.0	-	4.87	-
13.4	-	469.3	-3.4	-	4.95	-
15.1	-	451.5	-17.8	-	4.74	-
16.8	-	409.5	-32.2	-	4.25	11.8
18.5	-	343.4	-46.6	-	3.50	11.8
20.1	-	253.1	-61.0	-	2.53	11.8
21.8	-	138.6	-73.0	-	1.41	11.8
23.5	-	-	-73.0	-	-	11.8

B.2 Input and Output Files for Two Story Space Frame

B.2.1 Input File for Two Story Space Frame

```
STRUCTURE  Two story space frame - Test Problem 2
TYPE SPACE FRAME
METHOD STIFFNESS
TABULATE ALL
NUMBER OF JOINTS      18
NUMBER OF MEMBERS     26
NUMBER OF SUPPORTS     6
NUMBER OF LOADINGS     4
NUMBER OF LOAD COMBINATIONS     4
JOINT COORDINATES
  1     0.00     0.00     0.00 S
  2     0.00   144.00     0.00
  3     0.00   264.00     0.00
  4   360.00     0.00     0.00 S
  5   360.00   144.00     0.00
  6   360.00   264.00     0.00
  7   720.00     0.00     0.00 S
  8   720.00   144.00     0.00
  9   720.00   264.00     0.00
  10    0.00     0.00   360.00 S
  11    0.00   144.00   360.00
  12    0.00   264.00   360.00
  13  360.00     0.00   360.00 S
  14  360.00   144.00   360.00
  15  360.00   264.00   360.00
  16  720.00     0.00   360.00 S
  17  720.00   144.00   360.00
  18  720.00   264.00   360.00
MEMBER INCIDENCES
  1    1    2
  2    2    3
  3    4    5
  4    5    6
  5    7    8
  6    8    9
  7    2    5
  8    3    6
  9    5    8
 10    6    9
 11    2   11
 12    3   12
 13    5   14
 14    6   15
 15    8   17
 16    9   18
 17   10   11
 18   11   12
 19   13   14
 20   14   15
 21   16   17
 22   17   18
```

```
 23  11  14
 24  12  15
 25  14  17
 26  15  18
MEMBER PROPERTIES PRISMATIC
  1 AX IX IY IZ  324.00  0.00  8748.00  8748.00
  2 AX IX IY IZ  324.00  0.00  8748.00  8748.00
  3 AX IX IY IZ  324.00  0.00  8748.00  8748.00
  4 AX IX IY IZ  324.00  0.00  8748.00  8748.00
  5 AX IX IY IZ  324.00  0.00  8748.00  8748.00
  6 AX IX IY IZ  324.00  0.00  8748.00  8748.00
  7 AX IX IY IZ  504.00  0.00 13608.00 32928.00
  8 AX IX IY IZ  504.00  0.00 13608.00 32928.00
  9 AX IX IY IZ  504.00  0.00 13608.00 32928.00
 10 AX IX IY IZ  504.00  0.00 13608.00 32928.00
 11 AX IX IY IZ  240.00  0.00  2880.00  8000.00
 12 AX IX IY IZ  240.00  0.00  2880.00  8000.00
 13 AX IX IY IZ  384.00  0.00  8192.00 18432.00
 14 AX IX IY IZ  384.00  0.00  8192.00 18432.00
 15 AX IX IY IZ  240.00  0.00  2880.00  8000.00
 16 AX IX IY IZ  240.00  0.00  2880.00  8000.00
 17 AX IX IY IZ  324.00  0.00  8748.00  8748.00
 18 AX IX IY IZ  324.00  0.00  8748.00  8748.00
 19 AX IX IY IZ  324.00  0.00  8748.00  8748.00
 20 AX IX IY IZ  324.00  0.00  8748.00  8748.00
 21 AX IX IY IZ  324.00  0.00  8748.00  8748.00
 22 AX IX IY IZ  324.00  0.00  8748.00  8748.00
 23 AX IX IY IZ  504.00  0.00 13608.00 32928.00
 24 AX IX IY IZ  504.00  0.00 13608.00 32928.00
 25 AX IX IY IZ  504.00  0.00 13608.00 32928.00
 26 AX IX IY IZ  504.00  0.00 13608.00 32928.00
CONSTANTS E 201.39 ALL
LOADING 1 - dead load
MEMBER LOADS
  7 FORCE Y CONCENTRATED -23.5 120.
  7 FORCE Y CONCENTRATED -23.5 240.
  8 FORCE Y CONCENTRATED -23.5 120.
  8 FORCE Y CONCENTRATED -23.5 240.
  9 FORCE Y CONCENTRATED -23.5 120.
  9 FORCE Y CONCENTRATED -23.5 240.
 10 FORCE Y CONCENTRATED -23.5 120.
 10 FORCE Y CONCENTRATED -23.5 240.
 23 FORCE Y CONCENTRATED -23.5 120.
 23 FORCE Y CONCENTRATED -23.5 240.
 24 FORCE Y CONCENTRATED -23.5 120.
 24 FORCE Y CONCENTRATED -23.5 240.
 25 FORCE Y CONCENTRATED -23.5 120.
 25 FORCE Y CONCENTRATED -23.5 240.
 26 FORCE Y CONCENTRATED -23.5 120.
 26 FORCE Y CONCENTRATED -23.5 240.
 11 FORCE Y UNIFORM -0.05
 12 FORCE Y UNIFORM -0.05
 13 FORCE Y UNIFORM -0.10
 14 FORCE Y UNIFORM -0.10
 15 FORCE Y UNIFORM -0.05
 16 FORCE Y UNIFORM -0.05
```

```
LOADING 2 - live load
MEMBER LOADS
  7 FORCE Y CONCENTRATED  -7.5 120.
  7 FORCE Y CONCENTRATED  -7.5 240.
  8 FORCE Y CONCENTRATED  -7.5 120.
  8 FORCE Y CONCENTRATED  -7.5 240.
  9 FORCE Y CONCENTRATED  -7.5 120.
  9 FORCE Y CONCENTRATED  -7.5 240.
 10 FORCE Y CONCENTRATED  -7.5 120.
 10 FORCE Y CONCENTRATED  -7.5 240.
 23 FORCE Y CONCENTRATED  -7.5 120.
 23 FORCE Y CONCENTRATED  -7.5 240.
 24 FORCE Y CONCENTRATED  -7.5 120.
 24 FORCE Y CONCENTRATED  -7.5 240.
 25 FORCE Y CONCENTRATED  -7.5 120.
 25 FORCE Y CONCENTRATED  -7.5 240.
 26 FORCE Y CONCENTRATED  -7.5 120.
 26 FORCE Y CONCENTRATED  -7.5 240.
 11 FORCE Y UNIFORM -0.0208
 12 FORCE Y UNIFORM -0.0208
 13 FORCE Y UNIFORM -0.0500
 14 FORCE Y UNIFORM -0.0500
 15 FORCE Y UNIFORM -0.0208
 16 FORCE Y UNIFORM -0.0208
LOADING 3 - wind load (front)
JOINT LOADS
 11 FORCE Z -4.125
 12 FORCE Z -1.875
 14 FORCE Z -8.250
 15 FORCE Z -3.750
 17 FORCE Z -4.125
 18 FORCE Z -1.875
LOADING 4 wind load (side)
JOINT LOADS
 11 FORCE X  4.125
 12 FORCE X  1.875
  2 FORCE X  4.125
  3 FORCE X  1.875
LOAD COMBINATION 1 - 1.4 * 1 + 1.7 * 2
MEMBER LOADS
  7 FORCE Y CONCENTRATED -45.65 120.
  7 FORCE Y CONCENTRATED -45.65 240.
  8 FORCE Y CONCENTRATED -45.65 120.
  8 FORCE Y CONCENTRATED -45.65 240.
  9 FORCE Y CONCENTRATED -45.65 120.
  9 FORCE Y CONCENTRATED -45.65 240.
 10 FORCE Y CONCENTRATED -45.65 120.
 10 FORCE Y CONCENTRATED -45.65 240.
 23 FORCE Y CONCENTRATED -45.65 120.
 23 FORCE Y CONCENTRATED -45.65 240.
 24 FORCE Y CONCENTRATED -45.65 120.
 24 FORCE Y CONCENTRATED -45.65 240.
 25 FORCE Y CONCENTRATED -45.65 120.
 25 FORCE Y CONCENTRATED -45.65 240.
 26 FORCE Y CONCENTRATED -45.65 120.
 26 FORCE Y CONCENTRATED -45.65 240.
```

```
 11 FORCE Y UNIFORM -0.1054
 12 FORCE Y UNIFORM -0.1054
 13 FORCE Y UNIFORM -0.225
 14 FORCE Y UNIFORM -0.225
 15 FORCE Y UNIFORM -0.1054
 16 FORCE Y UNIFORM -0.1054
LOAD COMBINATION 2 - 1.05 * 1 + 1.275 * 2 + 1.275 * 4
JOINT LOADS
 11 FORCE X  5.259
 12 FORCE X  2.391
  2 FORCE X  5.259
  3 FORCE X  2.391
MEMBER LOADS
  7 FORCE Y CONCENTRATED  -34.24 120.
  7 FORCE Y CONCENTRATED  -34.24 240.
  8 FORCE Y CONCENTRATED  -34.24 120.
  8 FORCE Y CONCENTRATED  -34.24 240.
  9 FORCE Y CONCENTRATED  -34.24 120.
  9 FORCE Y CONCENTRATED  -34.24 240.
 10 FORCE Y CONCENTRATED  -34.24 120.
 10 FORCE Y CONCENTRATED  -34.24 240.
 23 FORCE Y CONCENTRATED  -34.24 120.
 23 FORCE Y CONCENTRATED  -34.24 240.
 24 FORCE Y CONCENTRATED  -34.24 120.
 24 FORCE Y CONCENTRATED  -34.24 240.
 25 FORCE Y CONCENTRATED  -34.24 120.
 25 FORCE Y CONCENTRATED  -34.24 240.
 26 FORCE Y CONCENTRATED  -34.24 120.
 26 FORCE Y CONCENTRATED  -34.24 240.
 11 FORCE Y UNIFORM -0.0790
 12 FORCE Y UNIFORM -0.0790
 13 FORCE Y UNIFORM -0.1688
 14 FORCE Y UNIFORM -0.1688
 15 FORCE Y UNIFORM -0.0790
 16 FORCE Y UNIFORM -0.0790
LOAD COMBINATION 3 - 1.05 * 1 + 1.275 * 2 + 1.275 * 3
JOINT LOADS
 11 FORCE Z  -5.259
 12 FORCE Z  -2.391
 14 FORCE Z -10.519
 15 FORCE Z  -4.781
 17 FORCE Z  -5.259
 18 FORCE Z  -2.391
MEMBER LOADS
  7 FORCE Y CONCENTRATED  -34.24 120.
  7 FORCE Y CONCENTRATED  -34.24 240.
  8 FORCE Y CONCENTRATED  -34.24 120.
  8 FORCE Y CONCENTRATED  -34.24 240.
  9 FORCE Y CONCENTRATED  -34.24 120.
  9 FORCE Y CONCENTRATED  -34.24 240.
 10 FORCE Y CONCENTRATED  -34.24 120.
 10 FORCE Y CONCENTRATED  -34.24 240.
 23 FORCE Y CONCENTRATED  -34.24 120.
 23 FORCE Y CONCENTRATED  -34.24 240.
 24 FORCE Y CONCENTRATED  -34.24 120.
 24 FORCE Y CONCENTRATED  -34.24 240.
```

```
 25 FORCE Y CONCENTRATED  -34.24 120.
 25 FORCE Y CONCENTRATED  -34.24 240.
 26 FORCE Y CONCENTRATED  -34.24 120.
 26 FORCE Y CONCENTRATED  -34.24 240.
 11 FORCE Y UNIFORM -0.0790
 12 FORCE Y UNIFORM -0.0790
 13 FORCE Y UNIFORM -0.1688
 14 FORCE Y UNIFORM -0.1688
 15 FORCE Y UNIFORM -0.0790
 16 FORCE Y UNIFORM -0.0790
LOAD COMBINATION 4 - 0.9 * 1 + 1.3 * 3
JOINT LOADS
 11 FORCE Z  -5.363
 12 FORCE Z  -2.438
 14 FORCE Z -10.725
 15 FORCE Z  -4.875
 17 FORCE Z  -5.363
 18 FORCE Z  -2.438
MEMBER LOADS
  7 FORCE Y CONCENTRATED  -21.15 120.
  7 FORCE Y CONCENTRATED  -21.15 240.
  8 FORCE Y CONCENTRATED  -21.15 120.
  8 FORCE Y CONCENTRATED  -21.15 240.
  9 FORCE Y CONCENTRATED  -21.15 120.
  9 FORCE Y CONCENTRATED  -21.15 240.
 10 FORCE Y CONCENTRATED  -21.15 120.
 10 FORCE Y CONCENTRATED  -21.15 240.
 23 FORCE Y CONCENTRATED  -21.15 120.
 23 FORCE Y CONCENTRATED  -21.15 240.
 24 FORCE Y CONCENTRATED  -21.15 120.
 24 FORCE Y CONCENTRATED  -21.15 240.
 25 FORCE Y CONCENTRATED  -21.15 120.
 25 FORCE Y CONCENTRATED  -21.15 240.
 26 FORCE Y CONCENTRATED  -21.15 120.
 26 FORCE Y CONCENTRATED  -21.15 240.
 11 FORCE Y UNIFORM -0.0450
 12 FORCE Y UNIFORM -0.0450
 13 FORCE Y UNIFORM -0.0900
 14 FORCE Y UNIFORM -0.0900
 15 FORCE Y UNIFORM -0.0450
 16 FORCE Y UNIFORM -0.0450
SOLVE

STRUCTURE 3-d Two story space frame
================================================================================

LOADING 1 - dead load
================================================================================

MEMBER FORCES

MEMB JOINT  AXIAL     SHEAR     SHEAR     TORSION     MOMENT      MOMENT
            FORCE   FORCE Y   FORCE Z    MOMENT         Y           Z
  1  1     58.85     -4.71      1.75      0.01     -86.01     -228.07
  1  2    -58.85      4.71     -1.75     -0.01     -166.28    -450.18
  2  2     28.79    -14.86      7.20      0.00     -365.23    -823.73
```

2	3	-28.79	14.86	-7.20	0.00	-498.79	-959.95
3	4	142.29	0.00	3.08	0.00	-148.60	0.00
3	5	-142.29	0.00	-3.08	0.00	-295.09	0.00
4	5	72.42	0.00	11.69	0.00	-619.69	0.00
4	6	-72.42	0.00	-11.69	0.00	-783.65	0.00
5	7	58.85	4.71	1.75	-0.01	-86.01	228.07
5	8	-58.85	-4.71	-1.75	0.01	-166.28	450.18
6	8	28.79	14.86	7.20	0.00	-365.23	823.73
6	9	-28.79	-14.86	-7.20	0.00	-498.79	959.95
7	2	-10.15	21.06	0.00	-20.88	0.00	1273.91
7	5	10.15	25.94	0.00	20.88	0.02	-2151.67
8	3	14.86	19.79	0.00	-28.87	0.01	959.95
8	6	-14.86	27.21	0.00	28.87	0.03	-2294.72
9	5	-10.15	25.94	0.00	20.88	-0.02	2151.67
9	8	10.15	21.06	0.00	-20.88	0.00	-1273.91
10	6	14.86	27.21	0.00	28.87	-0.03	2294.72
10	9	-14.86	19.79	0.00	-28.87	-0.01	-959.95
11	2	-5.45	9.00	0.00	0.00	0.00	510.64
11	11	5.45	9.00	0.00	0.00	0.00	-510.64
12	3	7.20	9.00	0.00	0.00	0.00	469.92
12	12	-7.20	9.00	0.00	0.00	0.00	-469.92
13	5	-8.61	18.00	0.00	0.00	0.00	956.54
13	14	8.61	18.00	0.00	0.00	0.00	-956.54
14	6	11.69	18.00	0.00	0.00	0.00	841.38
14	15	-11.69	18.00	0.00	0.00	0.00	-841.38
15	8	-5.45	9.00	0.00	0.00	0.00	510.64
15	17	5.45	9.00	0.00	0.00	0.00	-510.64
16	9	7.20	9.00	0.00	0.00	0.00	469.92
16	18	-7.20	9.00	0.00	0.00	0.00	-469.92
17	10	58.85	-4.71	-1.75	-0.01	86.01	-228.07
17	11	-58.85	4.71	1.75	0.01	166.28	-450.18
18	11	28.79	-14.86	-7.20	0.00	365.23	-823.73
18	12	-28.79	14.86	7.20	0.00	498.79	-959.95
19	13	142.29	0.00	-3.08	0.00	148.60	0.00
19	14	-142.29	0.00	3.08	0.00	295.09	0.00
20	14	72.42	0.00	-11.69	0.00	619.69	0.00
20	15	-72.42	0.00	11.69	0.00	783.65	0.00
21	16	58.85	4.71	-1.75	0.01	86.01	228.07
21	17	-58.85	-4.71	1.75	-0.01	166.28	450.18
22	17	28.79	14.86	-7.20	0.00	365.23	823.73
22	18	-28.79	-14.86	7.20	0.00	498.79	959.95
23	11	-10.15	21.06	0.00	20.88	0.00	1273.91
23	14	10.15	25.94	0.00	-20.88	-0.02	-2151.67
24	12	14.86	19.79	0.00	28.87	-0.01	959.95
24	15	-14.86	27.21	0.00	-28.87	-0.03	-2294.72
25	14	-10.15	25.94	0.00	-20.88	0.02	2151.67
25	17	10.15	21.06	0.00	20.88	0.00	-1273.91
26	15	14.86	27.21	0.00	-28.87	0.03	2294.72
26	18	-14.86	19.79	0.00	28.87	0.01	-959.95

FREE JOINT DISPLACEMENTS

JOINT	X DISPL	Y DISPL	Z DISPL	X-ROTAT	Y-ROTAT	Z-ROTAT
2	-.3692E-01	-.1467E+00	-.2765E-01	0.4288E-02	0.8323E-04	-0.8777E-02
3	0.5372E-01	-.2066E+00	0.3653E-01	0.1013E-01	-0.1097E-03	-0.1337E-01
5	-.4904E-14	-.3301E+00	-.4710E-01	0.7317E-02	-0.2201E-04	0.8594E-17

6	-.6226E-14	-.4699E+00	0.6222E-01	0.1729E-01	0.2863E-04	0.2137E-17
8	0.3692E-01	-.1467E+00	-.1532E-01	0.4177E-02	-0.9334E-04	0.8777E-02
9	-.5372E-01	-.2066E+00	0.2053E-01	0.8423E-02	0.1225E-03	0.1337E-01
11	-.3692E-01	-.1467E+00	0.2765E-01	-.4288E-02	-0.8323E-04	-0.8777E-02
12	0.5372E-01	-.2066E+00	-.3653E-01	-.1013E-01	0.1097E-03	-0.1337E-01
14	-.6865E-14	-.3301E+00	0.4710E-01	-.7317E-02	0.2201E-04	0.1041E-16
15	-.7853E-14	-.4699E+00	-.6222E-01	-.1729E-01	-0.2863E-04	0.1763E-17
17	0.3692E-01	-.1467E+00	0.1532E-01	-.4177E-02	0.9334E-04	0.8777E-02
18	-.5372E-01	-.2066E+00	-.2053E-01	-.8423E-02	-0.1225E-03	0.1337E-01

STRUCTURE 3-d Two story space frame

LOADING 2 - live load

MEMBER FORCES

MEMB	JOINT	AXIAL FORCE	SHEAR FORCE Y	SHEAR FORCE Z	TORSION MOMENT	MOMENT Y	MOMENT Z
1	1	20.63	-1.51	0.74	-0.01	-36.50	-73.13
1	2	-20.63	1.51	-0.74	0.01	-70.51	-144.33
2	2	10.11	-4.77	3.09	0.07	-156.14	-264.37
2	3	-10.11	4.77	-3.09	-0.07	-214.48	-308.40
3	4	51.87	0.00	1.53	0.00	-73.97	0.00
3	5	-51.87	0.00	-1.53	0.00	-146.93	0.00
4	5	26.34	0.00	5.78	0.00	-307.00	0.00
4	6	-26.34	0.00	-5.78	0.00	-386.72	0.00
5	7	20.63	1.51	0.74	0.01	-36.50	73.13
5	8	-20.63	-1.51	-0.74	-0.01	-70.51	144.33
6	8	10.11	4.77	3.09	-0.07	-156.14	264.37
6	9	-10.11	-4.77	-3.09	0.07	-214.48	308.40
7	2	-3.26	6.74	0.00	-12.28	-0.09	408.70
7	5	3.26	8.26	0.00	12.28	-0.12	-684.09
8	3	4.77	6.33	0.00	-17.90	0.08	308.40
8	6	-4.77	8.67	0.00	17.90	0.17	-729.15
9	5	-3.26	8.26	0.00	12.28	0.12	684.09
9	8	3.26	6.74	0.00	-12.28	0.09	-408.70
10	6	4.77	8.67	0.00	17.90	-0.17	729.15
10	9	-4.77	6.33	0.00	-17.90	-0.08	-308.40
11	2	-2.35	3.78	0.00	0.00	0.01	214.36
11	11	2.35	3.78	0.00	0.00	-0.01	-214.36
12	3	3.09	3.78	0.00	0.00	-0.02	196.58
12	12	-3.09	3.78	0.00	0.00	0.02	-196.58
13	5	-4.25	9.00	0.00	0.00	0.00	478.50
13	14	4.25	9.00	0.00	0.00	0.00	-478.50
14	6	5.78	9.00	0.00	0.00	0.00	422.52
14	15	-5.78	9.00	0.00	0.00	0.00	-422.52
15	8	-2.35	3.78	0.00	0.00	-0.01	214.36
15	17	2.35	3.78	0.00	0.00	0.01	-214.36
16	9	3.09	3.78	0.00	0.00	0.02	196.58
16	18	-3.09	3.78	0.00	0.00	-0.02	-196.58
17	10	20.63	-1.51	-0.74	0.01	36.50	-73.13
17	11	-20.63	1.51	0.74	-0.01	70.51	-144.33
18	11	10.11	-4.77	-3.09	-0.07	156.14	-264.37
18	12	-10.11	4.77	3.09	0.07	214.48	-308.40

19	13	51.87	0.00	-1.53	0.00	73.97	0.00
19	14	-51.87	0.00	1.53	0.00	146.93	0.00
20	14	26.34	0.00	-5.78	0.00	307.00	0.00
20	15	-26.34	0.00	5.78	0.00	386.72	0.00
21	16	20.63	1.51	-0.74	-0.01	36.50	73.13
21	17	-20.63	-1.51	0.74	0.01	70.51	144.33
22	17	10.11	4.77	-3.09	0.07	156.14	264.37
22	18	-10.11	-4.77	3.09	-0.07	214.48	308.40
23	11	-3.26	6.74	0.00	12.28	0.09	408.70
23	14	3.26	8.26	0.00	-12.28	0.12	-684.09
24	12	4.77	6.33	0.00	17.90	-0.08	308.40
24	15	-4.77	8.67	0.00	-17.90	-0.17	-729.15
25	14	-3.26	8.26	0.00	-12.28	-0.12	684.09
25	17	3.26	6.74	0.00	12.28	-0.09	-408.70
26	15	4.77	8.67	0.00	-17.90	0.17	729.15
26	18	-4.77	6.33	0.00	17.90	0.08	-308.40

FREE JOINT DISPLACEMENTS

JOINT	X DISPL	Y DISPL	Z DISPL	X-ROTAT	Y-ROTAT	Z-ROTAT
2	-.1187E-01	-.4544E-01	-.8129E-02	0.1260E-02	0.4616E-04	-0.2814E-02
3	0.1725E-01	-.6400E-01	0.1074E-01	0.2977E-02	-0.6091E-04	-0.4298E-02
5	-.1823E-14	-.1143E+00	-.1949E-01	0.3028E-02	-0.7415E-05	0.5810E-17
6	-.2661E-14	-.1627E+00	0.2575E-01	0.7156E-02	0.9662E-05	0.3368E-17
8	0.1187E-01	-.4544E-01	-.4503E-02	0.1227E-02	-0.4515E-04	0.2814E-02
9	-.1725E-01	-.6400E-01	0.6036E-02	0.2474E-02	0.5941E-04	0.4298E-02
11	-.1187E-01	-.4544E-01	0.8129E-02	-.1260E-02	-0.4616E-04	-0.2814E-02
12	0.1725E-01	-.6400E-01	-.1074E-01	-.2977E-02	0.6091E-04	-0.4298E-02
14	-.2331E-14	-.1143E+00	0.1949E-01	-.3028E-02	0.7415E-05	0.5821E-17
15	-.2987E-14	-.1627E+00	-.2575E-01	-.7156E-02	-0.9662E-05	0.3373E-17
17	0.1187E-01	-.4544E-01	0.4503E-02	-.1227E-02	0.4515E-04	0.2814E-02
18	-.1725E-01	-.6400E-01	-.6036E-02	-.2474E-02	-0.5941E-04	0.4298E-02

STRUCTURE 3-d Two story space frame

LOADING 3 - wind load (front)

MEMBER FORCES

MEMB	JOINT	AXIAL FORCE	SHEAR FORCE Y	SHEAR FORCE Z	TORSION MOMENT	MOMENT Y	MOMENT Z
1	1	1.50	0.00	3.17	-6.74	-307.31	-0.18
1	2	-1.50	0.00	-3.17	6.74	-149.88	-0.28
2	2	0.54	-0.01	1.06	-0.84	-23.35	-0.43
2	3	-0.54	0.01	-1.06	0.84	-103.82	-0.56
3	4	3.00	0.00	5.51	0.00	-472.86	0.00
3	5	-3.00	0.00	-5.51	0.00	-321.25	0.00
4	5	0.91	0.00	1.72	0.00	-53.05	0.00
4	6	-0.91	0.00	-1.72	0.00	-152.77	0.00
5	7	1.50	0.00	3.17	6.74	-307.31	0.18
5	8	-1.50	0.00	-3.17	-6.74	-149.88	0.28
6	8	0.54	0.01	1.06	0.84	-23.35	0.43
6	9	-0.54	-0.01	-1.06	-0.84	-103.82	0.56
7	2	-0.05	0.00	0.12	-1.83	-14.40	0.72

7	5	0.05	0.00	-0.12	1.83	-27.78	0.87
8	3	-0.04	0.00	0.10	-6.89	-10.20	0.59
8	6	0.04	0.00	-0.10	6.89	-24.97	0.91
9	5	-0.05	0.00	-0.12	1.83	27.78	-0.87
9	8	0.05	0.00	0.12	-1.83	14.40	-0.72
10	6	-0.04	0.00	-0.10	6.89	24.97	-0.91
10	9	0.04	0.00	0.10	-6.89	10.20	-0.59
11	2	2.00	0.95	-0.05	-0.01	8.50	171.40
11	11	-2.00	-0.95	0.05	0.01	8.54	171.76
12	3	0.96	0.54	-0.05	-0.03	9.37	96.94
12	12	-0.96	-0.54	0.05	0.03	9.39	96.04
13	5	4.03	2.10	0.00	0.00	0.00	377.97
13	14	-4.03	-2.10	0.00	0.00	0.00	378.82
14	6	1.91	0.92	0.00	0.00	0.00	166.54
14	15	-1.91	-0.92	0.00	0.00	0.00	164.41
15	8	2.00	0.95	0.05	0.01	-8.50	171.40
15	17	-2.00	-0.95	-0.05	-0.01	-8.54	171.76
16	9	0.96	0.54	0.05	0.03	-9.37	96.94
16	18	-0.96	-0.54	-0.05	-0.03	-9.39	96.04
17	10	-1.50	0.00	3.26	-6.83	-313.68	0.18
17	11	1.50	0.00	-3.26	6.83	-155.25	0.28
18	11	-0.54	0.01	1.01	-0.80	-18.35	0.43
18	12	0.54	-0.01	-1.01	0.80	-102.96	0.56
19	13	-3.00	0.00	5.62	0.00	-481.14	0.00
19	14	3.00	0.00	-5.62	0.00	-328.52	0.00
20	14	-0.91	0.00	1.64	0.00	-46.63	0.00
20	15	0.91	0.00	-1.64	0.00	-150.58	0.00
21	16	-1.50	0.00	3.26	6.83	-313.68	-0.18
21	17	1.50	0.00	-3.26	-6.83	-155.25	-0.28
22	17	-0.54	-0.01	1.01	0.80	-18.35	-0.43
22	18	0.54	0.01	-1.01	-0.80	-102.96	-0.56
23	11	0.05	0.00	0.12	-1.83	-14.56	-0.72
23	14	-0.05	0.00	-0.12	1.83	-28.11	-0.87
24	12	0.04	0.00	0.10	-6.91	-10.19	-0.59
24	15	-0.04	0.00	-0.10	6.91	-25.07	-0.91
25	14	0.05	0.00	-0.12	1.83	28.11	0.87
25	17	-0.05	0.00	0.12	-1.83	14.56	0.72
26	15	0.04	0.00	-0.10	6.91	25.07	0.91
26	18	-0.04	0.00	0.10	-6.91	10.19	0.59

FREE JOINT DISPLACEMENTS

JOINT	X DISPL	Y DISPL	Z DISPL	X-ROTAT	Y-ROTAT	Z-ROTAT
2	0.2118E-01	-.3576E-02	-.9850E+00	-.7062E-02	0.1348E-02	-0.9701E-04
3	0.3448E-01	-.4698E-02	-.1767E+01	-.4303E-02	0.2193E-02	-0.4870E-04
5	0.2163E-01	-.4799E-02	-.1376E+01	-.9465E-02	-0.3819E-03	-0.5202E-04
6	0.3514E-01	-.6204E-02	-.2366E+01	-.5275E-02	-0.9147E-03	-0.2523E-04
8	0.2208E-01	-.4630E-02	-.7975E+00	-.4104E-02	-0.1215E-02	-0.1001E-03
9	0.3604E-01	-.5817E-02	-.1265E+01	-.1870E-02	-0.2261E-02	-0.4727E-04
11	-.2118E-01	0.3576E-02	-.9999E+00	-.7104E-02	0.1409E-02	0.9701E-04
12	-.3448E-01	0.4698E-02	-.1774E+01	-.4201E-02	0.2222E-02	0.4870E-04
14	-.2163E-01	0.4799E-02	-.1406E+01	-.9549E-02	-0.3927E-03	0.5202E-04
15	-.3514E-01	0.6204E-02	-.2380E+01	-.5072E-02	-0.9200E-03	0.2523E-04
17	-.2208E-01	0.4630E-02	-.8069E+00	-.4125E-02	-0.1276E-02	0.1001E-03
18	-.3604E-01	0.5817E-02	-.1269E+01	-.1817E-02	-0.2291E-02	0.4727E-04

STRUCTURE 3-d Two story space frame

LOADING 4 wind load (side)

MEMBER FORCES

MEMB	JOINT	AXIAL FORCE	SHEAR FORCE Y	SHEAR FORCE Z	TORSION MOMENT	MOMENT Y	MOMENT Z
1	1	-0.85	1.93	0.00	0.00	0.00	156.70
1	2	0.85	-1.93	0.00	0.00	0.00	121.61
2	2	-0.20	0.39	0.00	0.00	0.00	8.27
2	3	0.20	-0.39	0.00	0.00	0.00	38.49
3	4	-0.01	2.22	0.00	0.00	0.00	168.79
3	5	0.01	-2.22	0.00	0.00	0.00	151.22
4	5	-0.01	1.03	0.00	0.00	0.00	55.17
4	6	0.01	-1.03	0.00	0.00	0.00	68.39
5	7	0.86	1.85	0.00	0.00	0.00	150.17
5	8	-0.86	-1.85	0.00	0.00	0.00	115.51
6	8	0.21	0.46	0.00	0.00	0.00	13.11
6	9	-0.21	-0.46	0.00	0.00	0.00	41.57
7	2	2.58	-0.65	0.00	0.00	0.00	-129.89
7	5	-2.58	0.65	0.00	0.00	0.00	-103.51
8	3	1.49	-0.20	0.00	0.00	0.00	-38.49
8	6	-1.49	0.20	0.00	0.00	0.00	-33.44
9	5	1.39	-0.64	0.00	0.00	0.00	-102.87
9	8	-1.39	0.64	0.00	0.00	0.00	-128.61
10	6	0.46	-0.21	0.00	0.00	0.00	-34.95
10	9	-0.46	0.21	0.00	0.00	0.00	-41.57
11	2	0.00	0.00	0.00	0.00	0.00	0.00
11	11	0.00	0.00	0.00	0.00	0.00	0.00
12	3	0.00	0.00	0.00	0.00	0.00	0.00
12	12	0.00	0.00	0.00	0.00	0.00	0.00
13	5	0.00	0.00	0.00	0.00	0.00	0.00
13	14	0.00	0.00	0.00	0.00	0.00	0.00
14	6	0.00	0.00	0.00	0.00	0.00	0.00
14	15	0.00	0.00	0.00	0.00	0.00	0.00
15	8	0.00	0.00	0.00	0.00	0.00	0.00
15	17	0.00	0.00	0.00	0.00	0.00	0.00
16	9	0.00	0.00	0.00	0.00	0.00	0.00
16	18	0.00	0.00	0.00	0.00	0.00	0.00
17	10	-0.85	1.93	0.00	0.00	0.00	156.70
17	11	0.85	-1.93	0.00	0.00	0.00	121.61
18	11	-0.20	0.39	0.00	0.00	0.00	8.27
18	12	0.20	-0.39	0.00	0.00	0.00	38.49
19	13	-0.01	2.22	0.00	0.00	0.00	168.79
19	14	0.01	-2.22	0.00	0.00	0.00	151.22
20	14	-0.01	1.03	0.00	0.00	0.00	55.17
20	15	0.01	-1.03	0.00	0.00	0.00	68.39
21	16	0.86	1.85	0.00	0.00	0.00	150.17
21	17	-0.86	-1.85	0.00	0.00	0.00	115.51
22	17	0.21	0.46	0.00	0.00	0.00	13.11
22	18	-0.21	-0.46	0.00	0.00	0.00	41.57
23	11	2.58	-0.65	0.00	0.00	0.00	-129.89
23	14	-2.58	0.65	0.00	0.00	0.00	-103.51

24	12	1.49	-0.20	0.00	0.00	0.00	-38.49
24	15	-1.49	0.20	0.00	0.00	0.00	-33.44
25	14	1.39	-0.64	0.00	0.00	0.00	-102.87
25	17	-1.39	0.64	0.00	0.00	0.00	-128.61
26	15	0.46	-0.21	0.00	0.00	0.00	-34.95
26	18	-0.46	0.21	0.00	0.00	0.00	-41.57

FREE JOINT DISPLACEMENTS

JOINT	X DISPL	Y DISPL	Z DISPL	X-ROTAT	Y-ROTAT	Z-ROTAT
2	-.4627E-01	0.5118E-02	0.1531E+01	0.1061E-01	0.3311E-02	0.1990E-03
3	-.7334E-01	0.6661E-02	0.2651E+01	0.5973E-02	0.5539E-02	0.8278E-04
5	-.4605E-01	0.1114E-02	0.2504E+00	0.1954E-02	0.2115E-02	0.1006E-03
6	-.7281E-01	0.1499E-02	0.4897E+00	0.1385E-02	0.3675E-02	0.4020E-04
8	-.4566E-01	-.2093E-04	-.4923E-01	-.2761E-03	0.2207E-03	0.1999E-03
9	-.7244E-01	-.3714E-04	-.8477E-01	-.1492E-03	0.4833E-03	0.8940E-04
11	0.4627E-01	-.5118E-02	0.1531E+01	0.1061E-01	0.3311E-02	-0.1990E-03
12	0.7334E-01	-.6661E-02	0.2651E+01	0.5973E-02	0.5539E-02	-0.8278E-04
14	0.4605E-01	-.1114E-02	0.2504E+00	0.1954E-02	0.2115E-02	-0.1006E-03
15	0.7281E-01	-.1499E-02	0.4897E+00	0.1385E-02	0.3675E-02	-0.4020E-04
17	0.4566E-01	0.2093E-04	-.4923E-01	-.2761E-03	0.2207E-03	-0.1999E-03
18	0.7244E-01	0.3714E-04	-.8477E-01	-.1492E-03	0.4833E-03	-0.8940E-04

STRUCTURE 3-d Two story space frame

==========

LOAD COMBINATION 1 - 1.4 * 1 + 1.7 * 2

==========

MEMBER FORCES

MEMB	JOINT	AXIAL FORCE	SHEAR FORCE Y	SHEAR FORCE Z	TORSION MOMENT	MOMENT Y	MOMENT Z
1	1	117.46	-9.16	3.72	-0.02	-182.46	-443.62
1	2	-117.46	9.16	-3.72	0.02	-352.66	-875.62
2	2	57.50	-28.92	15.33	0.12	-776.76	-1602.65
2	3	-57.50	28.92	-15.33	-0.12	-1062.92	-1868.21
3	4	287.38	0.00	6.92	0.00	-333.78	0.00
3	5	-287.38	0.00	-6.92	0.00	-662.91	0.00
4	5	146.16	0.00	26.20	0.00	-1389.47	0.00
4	6	-146.16	0.00	-26.20	0.00	-1754.53	0.00
5	7	117.46	9.16	3.72	0.02	-182.46	443.62
5	8	-117.46	-9.16	-3.72	-0.02	-352.66	875.62
6	8	57.50	28.92	15.33	-0.12	-776.76	1602.65
6	9	-57.50	-28.92	-15.33	0.12	-1062.92	1868.21
7	2	-19.76	40.94	0.00	-50.11	-0.14	2478.27
7	5	19.76	50.36	0.00	50.11	-0.17	-4175.30
8	3	28.92	38.47	0.00	-70.84	0.16	1868.21
8	6	-28.92	52.83	0.00	70.84	0.32	-4452.16
9	5	-19.76	50.36	0.00	50.11	0.17	4175.30
9	8	19.76	40.94	0.00	-50.11	0.14	-2478.27
10	6	28.92	52.83	0.00	70.84	-0.32	4452.16
10	9	-28.92	38.47	0.00	-70.84	-0.16	-1868.21
11	2	-11.62	19.03	0.00	0.00	0.01	1079.30
11	11	11.62	19.03	0.00	0.00	-0.01	-1079.30
12	3	15.33	19.03	0.00	0.00	-0.04	992.07

```
 12 12    -15.33     19.03      0.00      0.00      0.04    -992.07
 13  5    -19.28     40.50      0.00      0.00      0.00    2152.61
 13 14     19.28     40.50      0.00      0.00      0.00   -2152.61
 14  6     26.20     40.50      0.00      0.00      0.00    1896.22
 14 15    -26.20     40.50      0.00      0.00      0.00   -1896.22
 15  8    -11.62     19.03      0.00      0.00     -0.01    1079.30
 15 17     11.62     19.03      0.00      0.00      0.01   -1079.30
 16  9     15.33     19.03      0.00      0.00      0.04     992.07
 16 18    -15.33     19.03      0.00      0.00     -0.04    -992.07
 17 10    117.46     -9.16     -3.72      0.02    182.46    -443.62
 17 11   -117.46      9.16      3.72     -0.02    352.66    -875.62
 18 11     57.50    -28.92    -15.33     -0.12    776.76   -1602.65
 18 12    -57.50     28.92     15.33      0.12   1062.92   -1868.21
 19 13    287.38      0.00     -6.92      0.00    333.78       0.00
 19 14   -287.38      0.00      6.92      0.00    662.91       0.00
 20 14    146.16      0.00    -26.20      0.00   1389.47       0.00
 20 15   -146.16      0.00     26.20      0.00   1754.53       0.00
 21 16    117.46      9.16     -3.72     -0.02    182.46     443.62
 21 17   -117.46     -9.16      3.72      0.02    352.66     875.62
 22 17     57.50     28.92    -15.33      0.12    776.76    1602.65
 22 18    -57.50    -28.92     15.33     -0.12   1062.92    1868.21
 23 11    -19.76     40.94      0.00     50.11      0.14    2478.27
 23 14     19.76     50.36      0.00    -50.11      0.17   -4175.30
 24 12     28.92     38.47      0.00     70.84     -0.16    1868.21
 24 15    -28.92     52.83      0.00    -70.84     -0.32   -4452.16
 25 14    -19.76     50.36      0.00    -50.11     -0.17    4175.30
 25 17     19.76     40.94      0.00     50.11     -0.14   -2478.27
 26 15     28.92     52.83      0.00    -70.84      0.32    4452.16
 26 18    -28.92     38.47      0.00     70.84      0.16   -1868.21

FREE JOINT DISPLACEMENTS

JOINT X DISPL   Y DISPL   Z DISPL   X-ROTAT   Y-ROTAT     Z-ROTAT
  2 -.7186E-01 -.2826E+00 -.5254E-01 0.8147E-02  0.1949E-03 -0.1707E-01
  3 0.1045E+00 -.3980E+00 0.6941E-01 0.1925E-01 -0.2571E-03 -0.2602E-01
  5 -.9696E-14 -.6564E+00 -.9907E-01 0.1539E-01 -0.4343E-04  0.2268E-16
  6 -.1288E-13 -.9344E+00 0.1309E+00 0.3637E-01  0.5651E-04  0.1103E-16
  8 0.7186E-01 -.2826E+00 -.2910E-01 0.7935E-02 -0.2074E-03  0.1707E-01
  9 -.1045E+00 -.3980E+00 0.3901E-01 0.1600E-01  0.2725E-03  0.2602E-01
 11 -.7186E-01 -.2826E+00 0.5254E-01 -.8147E-02 -0.1949E-03 -0.1707E-01
 12 0.1045E+00 -.3980E+00 -.6941E-01 -.1925E-01  0.2571E-03 -0.2602E-01
 14 -.1396E-13 -.6564E+00 0.9907E-01 -.1539E-01  0.4343E-04  0.2768E-16
 15 -.1692E-13 -.9344E+00 -.1309E+00 -.3637E-01 -0.5651E-04  0.1134E-16
 17 0.7186E-01 -.2826E+00 0.2910E-01 -.7935E-02  0.2074E-03  0.1707E-01
 18 -.1045E+00 -.3980E+00 -.3901E-01 -.1600E-01 -0.2725E-03  0.2602E-01

STRUCTURE 3-d Two story space frame
================================================================================

LOAD COMBINATION 2 - 1.05 * 1 + 1.275 * 2 + 1.275 * 4
================================================================================

MEMBER FORCES

MEMB JOINT  AXIAL     SHEAR     SHEAR     TORSION     MOMENT      MOMENT
            FORCE    FORCE Y   FORCE Z    MOMENT        Y           Z
```

1	1	87.01	-4.41	2.79	-0.01	-136.85	-132.92
1	2	-87.01	4.41	-2.79	0.01	-264.49	-501.66
2	2	42.87	-21.20	11.50	0.09	-582.57	-1191.44
2	3	-42.87	21.20	-11.50	-0.09	-797.19	-1352.08
3	4	215.53	2.83	5.19	0.00	-250.34	215.20
3	5	-215.53	-2.83	-5.19	0.00	-497.19	192.80
4	5	109.60	1.31	19.65	0.00	-1042.10	70.34
4	6	-109.60	-1.31	-19.65	0.00	-1315.90	87.20
5	7	89.19	9.22	2.79	0.01	-136.85	524.18
5	8	-89.19	-9.22	-2.79	-0.01	-264.49	803.99
6	8	43.39	22.27	11.50	-0.09	-582.57	1218.70
6	9	-43.39	-22.27	-11.50	0.09	-797.19	1454.16
7	2	-11.53	29.88	0.00	-37.58	-0.11	1693.10
7	5	11.53	38.60	0.00	37.58	-0.13	-3263.45
8	3	23.59	28.60	0.00	-53.13	0.12	1352.08
8	6	-23.59	39.88	0.00	53.13	0.24	-3381.76
9	5	-13.05	36.95	0.00	37.58	0.13	3000.31
9	8	13.05	31.52	0.00	-37.58	0.11	-2022.68
10	6	22.27	39.35	0.00	53.13	-0.24	3294.56
10	9	-22.27	29.13	0.00	-53.13	-0.12	-1454.16
11	2	-8.71	14.27	0.00	0.00	0.00	809.48
11	11	8.71	14.27	0.00	0.00	0.00	-809.48
12	3	11.50	14.27	0.00	0.00	-0.03	744.06
12	12	-11.50	14.27	0.00	0.00	0.03	-744.05
13	5	-14.46	30.38	0.00	0.00	0.00	1614.46
13	14	14.46	30.38	0.00	0.00	0.00	-1614.46
14	6	19.65	30.38	0.00	0.00	0.00	1422.16
14	15	-19.65	30.38	0.00	0.00	0.00	-1422.16
15	8	-8.71	14.27	0.00	0.00	0.00	809.48
15	17	8.71	14.27	0.00	0.00	0.00	-809.48
16	9	11.50	14.27	0.00	0.00	0.03	744.06
16	18	-11.50	14.27	0.00	0.00	-0.03	-744.06
17	10	87.01	-4.41	-2.79	0.01	136.85	-132.92
17	11	-87.01	4.41	2.79	-0.01	264.49	-501.66
18	11	42.87	-21.20	-11.50	-0.09	582.57	-1191.44
18	12	-42.87	21.20	11.50	0.09	797.19	-1352.08
19	13	215.53	2.83	-5.19	0.00	250.34	215.20
19	14	-215.53	-2.83	5.19	0.00	497.19	192.80
20	14	109.60	1.31	-19.65	0.00	1042.10	70.34
20	15	-109.60	-1.31	19.65	0.00	1315.90	87.20
21	16	89.19	9.22	-2.79	-0.01	136.85	524.18
21	17	-89.19	-9.22	2.79	0.01	264.49	803.99
22	17	43.39	22.27	-11.50	0.09	582.57	1218.70
22	18	-43.39	-22.27	11.50	-0.09	797.19	1454.16
23	11	-11.53	29.88	0.00	37.58	0.11	1693.10
23	14	11.53	38.60	0.00	-37.58	0.13	-3263.45
24	12	23.59	28.60	0.00	53.13	-0.12	1352.08
24	15	-23.59	39.88	0.00	-53.13	-0.24	-3381.76
25	14	-13.05	36.95	0.00	-37.58	-0.13	3000.31
25	17	13.05	31.52	0.00	37.58	-0.11	-2022.68
26	15	22.27	39.35	0.00	-53.13	0.24	3294.56
26	18	-22.27	29.13	0.00	53.13	0.12	-1454.16

FREE JOINT DISPLACEMENTS

JOINT	X DISPL	Y DISPL	Z DISPL	X-ROTAT	Y-ROTAT	Z-ROTAT

2	-.1129E+00	-.2054E+00	0.1913E+01	0.1965E-01	0.4367E-02	-0.1255E-01
3	-.1511E-01	-.2901E+00	0.3432E+01	0.2206E-01	0.6869E-02	-0.1941E-01
5	-.5872E-01	-.4909E+00	0.2449E+00	0.1404E-01	0.2664E-02	0.1283E-03
6	-.9283E-01	-.6990E+00	0.7225E+00	0.2904E-01	0.4728E-02	0.5126E-04
8	-.4320E-02	-.2120E+00	-.8460E-01	0.5601E-02	0.1258E-03	0.1306E-01
9	-.1708E+00	-.2986E+00	-.7883E-01	0.1181E-01	0.8205E-03	0.1963E-01
11	0.5103E-02	-.2185E+00	0.1992E+01	0.7422E-02	0.4075E-02	-0.1306E-01
12	0.1719E+00	-.3071E+00	0.3328E+01	-.6827E-02	0.7255E-02	-0.1962E-01
14	0.5872E-01	-.4938E+00	0.3935E+00	-.9053E-02	0.2729E-02	-0.1283E-03
15	0.9283E-01	-.7028E+00	0.5262E+00	-.2551E-01	0.4643E-02	-0.5126E-04
17	0.1121E+00	-.2119E+00	-.4094E-01	-.6305E-02	0.4369E-03	0.1255E-01
18	0.1396E-01	-.2985E+00	-.1374E+00	-.1219E-01	0.4118E-03	0.1940E-01

STRUCTURE 3-d Two story space frame

LOAD COMBINATION 3 - 1.05 * 1 + 1.275 * 2 + 1.275 * 3

MEMBER FORCES

MEMB	JOINT	AXIAL FORCE	SHEAR FORCE Y	SHEAR FORCE Z	TORSION MOMENT	MOMENT Y	MOMENT Z
1	1	90.01	-6.88	6.84	-8.61	-528.67	-332.95
1	2	-90.01	6.88	-6.84	8.61	-455.59	-657.07
2	2	43.81	-21.70	12.85	-0.98	-612.34	-1202.53
2	3	-43.81	21.70	-12.85	0.98	-929.56	-1401.88
3	4	219.37	0.00	12.22	0.00	-853.23	0.00
3	5	-219.37	0.00	-12.22	0.00	-906.78	0.00
4	5	110.78	0.00	21.84	0.00	-1109.75	0.00
4	6	-110.78	0.00	-21.84	0.00	-1510.67	0.00
5	7	90.01	6.88	6.84	8.61	-528.67	332.95
5	8	-90.01	-6.88	-6.84	-8.61	-455.59	657.07
6	8	43.81	21.70	12.85	0.98	-612.34	1202.53
6	9	-43.81	-21.70	-12.85	-0.98	-929.56	1401.88
7	2	-14.89	30.71	0.15	-39.92	-18.47	1859.62
7	5	14.89	37.77	-0.15	39.92	-35.55	-3130.37
8	3	21.64	28.86	0.12	-61.92	-12.89	1401.91
8	6	-21.64	39.62	-0.12	61.92	-31.59	-3337.96
9	5	-14.89	37.77	-0.15	39.92	35.55	3130.37
9	8	14.89	30.71	0.15	-39.92	18.47	-1859.62
10	6	21.64	39.62	-0.12	61.92	31.59	3337.96
10	9	-21.64	28.86	0.12	-61.92	12.89	-1401.91
11	2	-6.16	15.48	-0.06	-0.02	10.84	1028.01
11	11	6.16	13.05	0.06	0.02	10.88	-590.48
12	3	12.73	14.95	-0.07	-0.04	11.92	867.65
12	12	-12.73	13.59	0.07	0.04	12.00	-621.60
13	5	-9.31	33.06	0.00	0.00	0.00	2096.37
13	14	9.31	27.69	0.00	0.00	0.00	-1131.46
14	6	22.08	31.55	0.00	0.00	0.00	1634.50
14	15	-22.08	29.20	0.00	0.00	0.00	-1212.53
15	8	-6.16	15.48	0.06	0.02	-10.84	1028.01
15	17	6.16	13.05	-0.06	-0.02	-10.88	-590.48
16	9	12.73	14.95	0.07	0.04	-11.92	867.65
16	18	-12.73	13.59	-0.07	-0.04	-12.00	-621.60
17	10	86.19	-6.87	1.36	-8.69	-263.09	-332.48

17	11	-86.19	6.87	-1.36	8.69	66.56	-656.36
18	11	42.43	-21.68	-10.21	-1.11	559.17	-1201.44
18	12	-42.43	21.68	10.21	1.11	665.92	-1400.44
19	13	211.71	0.00	1.98	0.00	-363.12	0.00
19	14	-211.71	0.00	-1.98	0.00	78.32	0.00
20	14	108.45	0.00	-17.55	0.00	982.64	0.00
20	15	-108.45	0.00	17.55	0.00	1123.90	0.00
21	16	86.19	6.87	1.36	8.69	-263.09	332.48
21	17	-86.19	-6.87	-1.36	-8.69	66.55	656.36
22	17	42.43	21.68	-10.21	1.11	559.17	1201.44
22	18	-42.43	-21.68	10.21	-1.11	665.92	1400.44
23	11	-14.75	30.70	0.15	35.25	-18.45	1857.78
23	14	14.75	37.78	-0.15	-35.25	-35.71	-3132.58
24	12	21.75	28.85	0.13	44.32	-13.11	1400.40
24	15	-21.75	39.63	-0.13	-44.32	-32.21	-3340.28
25	14	-14.75	37.78	-0.15	-35.25	35.71	3132.58
25	17	14.75	30.70	0.15	35.25	18.45	-1857.78
26	15	21.75	39.63	-0.13	-44.32	32.21	3340.27
26	18	-21.75	28.85	0.13	44.32	13.11	-1400.40

FREE JOINT DISPLACEMENTS

JOINT	X DISPL	Y DISPL	Z DISPL	X-ROTAT	Y-ROTAT	Z-ROTAT
2	-.2689E-01	-.2165E+00	-.1295E+01	-.2892E-02	0.1865E-02	-0.1293E-01
3	0.1224E+00	-.3046E+00	-.2201E+01	0.8956E-02	0.2603E-02	-0.1958E-01
5	0.2757E-01	-.4985E+00	-.1829E+01	-.5229E-03	-0.5195E-03	-0.6632E-04
6	0.4480E-01	-.7088E+00	-.2918E+01	0.2055E-01	-0.1124E-02	-0.3216E-04
8	0.8206E-01	-.2179E+00	-.1039E+01	0.7201E-03	-0.1705E-02	0.1268E-01
9	-.3245E-01	-.3060E+00	-.1583E+01	0.9618E-02	-0.2678E-02	0.1945E-01
11	-.8091E-01	-.2074E+00	-.1235E+01	-.1517E-01	0.1650E-02	-0.1268E-01
12	0.3444E-01	-.2926E+00	-.2314E+01	-.1980E-01	0.3026E-02	-0.1945E-01
14	-.2757E-01	-.4862E+00	-.1718E+01	-.2372E-01	-0.4681E-03	0.6632E-04
15	-.4480E-01	-.6930E+00	-.3133E+01	-.3374E-01	-0.1215E-02	0.3216E-04
17	0.2574E-01	-.2061E+00	-.1007E+01	-.1121E-01	-0.1471E-02	0.1293E-01
18	-.1244E+00	-.2912E+00	-.1648E+01	-.1432E-01	-0.3125E-02	0.1957E-01

STRUCTURE 3-d Two story space frame

LOAD COMBINATION 4 - 0.9 * 1 + 1.3 * 3

MEMBER FORCES

MEMB	JOINT	AXIAL FORCE	SHEAR FORCE Y	SHEAR FORCE Z	TORSION MOMENT	MOMENT Y	MOMENT Z
1	1	54.92	-4.24	5.70	-8.76	-476.91	-205.50
1	2	-54.92	4.24	-5.70	8.76	-344.50	-405.52
2	2	26.62	-13.39	7.86	-1.08	-359.06	-741.92
2	3	-26.62	13.39	-7.86	1.08	-583.88	-864.69
3	4	131.97	0.00	9.94	0.00	-748.46	0.00
3	5	-131.97	0.00	-9.94	0.00	-683.20	0.00
4	5	66.36	0.00	12.75	0.00	-626.69	0.00
4	6	-66.36	0.00	-12.75	0.00	-903.88	0.00
5	7	54.92	4.24	5.70	8.76	-476.91	205.50
5	8	-54.92	-4.24	-5.70	-8.76	-344.50	405.52

```
  6  8     26.62     13.39     7.86      1.08   -359.06     741.92
  6  9    -26.62    -13.39    -7.86     -1.08   -583.88     864.69
  7  2     -9.21     18.96     0.15    -21.17    -18.72    1147.46
  7  5      9.21     23.34    -0.15     21.17    -36.10   -1935.38
  8  3     13.32     17.82     0.13    -34.93    -13.26     864.73
  8  6    -13.32     24.48    -0.13     34.93    -32.43   -2064.07
  9  5     -9.21     23.34    -0.15     21.17     36.10    1935.38
  9  8      9.21     18.96     0.15    -21.17     18.72   -1147.46
 10  6     13.32     24.48    -0.13     34.93     32.43    2064.07
 10  9    -13.32     17.82     0.13    -34.93     13.26    -864.73
 11  2     -2.31      9.34    -0.06     -0.02     11.05     682.39
 11 11      2.31      6.86     0.06      0.02     11.10    -236.28
 12  3      7.73      8.80    -0.07     -0.04     12.17     548.95
 12 12     -7.73      7.40     0.07      0.04     12.21    -298.07
 13  5     -2.51     18.93     0.00      0.00      0.00    1352.24
 13 14      2.51     13.47     0.00      0.00      0.00    -368.42
 14  6     13.01     17.40     0.00      0.00      0.00     973.75
 14 15    -13.01     15.00     0.00      0.00      0.00    -543.51
 15  8     -2.31      9.34     0.06      0.02    -11.05     682.39
 15 17      2.31      6.86    -0.06     -0.02    -11.10    -236.28
 16  9      7.73      8.80     0.07      0.04    -12.17     548.95
 16 18     -7.73      7.40    -0.07     -0.04    -12.21    -298.07
 17 10     51.02     -4.23     2.66     -8.88   -330.38    -205.03
 17 11    -51.02      4.23    -2.66      8.88    -52.16    -404.80
 18 11     25.21    -13.37    -5.17     -1.05    304.85    -740.80
 18 12    -25.21     13.37     5.17      1.05    315.06    -863.22
 19 13    124.16      0.00     4.54      0.00   -491.74       0.00
 19 14   -124.16      0.00    -4.54      0.00   -161.49       0.00
 20 14     63.99      0.00    -8.39      0.00    497.10       0.00
 20 15    -63.99      0.00     8.39      0.00    509.53       0.00
 21 16     51.02      4.23     2.66      8.88   -330.38     205.03
 21 17    -51.02     -4.23    -2.66     -8.88    -52.16     404.80
 22 17     25.21     13.37    -5.17      1.05    304.85     740.80
 22 18    -25.21    -13.37     5.17     -1.05    315.06     863.22
 23 11     -9.07     18.95     0.15     16.41    -18.93    1145.58
 23 14      9.07     23.35    -0.15    -16.41    -36.56   -1937.63
 24 12     13.43     17.81     0.13     16.99    -13.26     863.18
 24 15    -13.43     24.49    -0.13    -16.99    -32.62   -2066.43
 25 14     -9.07     23.35    -0.15    -16.41     36.56    1937.63
 25 17      9.07     18.95     0.15     16.41     18.93   -1145.58
 26 15     13.43     24.49    -0.13    -16.99     32.62    2066.43
 26 18    -13.43     17.81     0.13     16.99     13.26    -863.18

FREE JOINT DISPLACEMENTS

JOINT X DISPL    Y DISPL    Z DISPL    X-ROTAT    Y-ROTAT    Z-ROTAT
  2 -.5690E-02 -.1366E+00 -.1306E+01 -.5323E-02  0.1828E-02 -0.8025E-02
  3 0.9317E-01 -.1920E+00 -.2264E+01 0.3524E-02  0.2752E-02 -0.1209E-01
  5 0.2811E-01 -.3033E+00 -.1831E+01 -.5720E-02 -0.5162E-03 -0.6762E-04
  6 0.4568E-01 -.4310E+00 -.3020E+01 0.8700E-02 -0.1163E-02 -0.3280E-04
  8 0.6194E-01 -.1380E+00 -.1051E+01 -.1578E-02 -0.1663E-02  0.7769E-02
  9 -.1490E-02 -.1935E+00 -.1626E+01 0.5146E-02 -0.2829E-02  0.1197E-01
 11 -.6077E-01 -.1273E+00 -.1275E+01 -.1309E-01  0.1757E-02 -0.7773E-02
 12 0.3514E-02 -.1798E+00 -.2339E+01 -.1458E-01  0.2987E-02 -0.1197E-01
 14 -.2811E-01 -.2908E+00 -.1785E+01 -.1900E-01 -0.4907E-03  0.6762E-04
 15 -.4568E-01 -.4148E+00 -.3150E+01 -.2215E-01 -0.1222E-02  0.3280E-04
```

```
17 0.4519E-02 -.1260E+00 -.1035E+01 -.9121E-02 -0.1574E-02  0.8029E-02
18 -.9520E-01 -.1783E+00 -.1669E+01 -.9940E-02 -0.3088E-02  0.1209E-01
```

B.2.2 Output File for Two Story Space Frame

Design results for structure: Two story space frame - Test Problem 2

Beam design results

Member # 7 type: Beam Independent member

width = 18.0 inches f'c = 4.0 ksi Stirrup size = # 4
Depth = 28.0 inches fy = 60.0 ksi

Distance From Left End (ft)	Design Moment Negative (kip-ft)	Design Moment Positive (kip-ft)	Design Shear Force (kip)	Area of steel Bottom (sq.in.)	Area of steel Top (sq.in.)	Stirrup Spacing (inch)
0.0	206.5	-	40.9	1.87	-	12.8
2.1	118.8	-	40.9	1.53	-	12.8
4.3	31.1	-	40.9	1.53	-	12.8
6.4	-	56.6	40.9	-	1.53	12.8
8.6	-	144.4	40.9	-	1.53	12.8
10.7	-	199.5	-4.7	-	1.80	-
12.9	-	189.4	-4.7	-	1.71	-
15.0	-	179.3	-4.7	-	1.61	-
17.1	-	169.2	-4.7	-	1.53	-
19.3	-	159.1	-4.7	-	1.53	-
21.4	-	83.7	-50.4	-	1.53	12.8
23.6	24.2	-	-50.4	1.53	-	12.8
25.7	132.1	-	-50.4	1.53	-	12.8
27.9	240.0	-	-50.4	2.18	-	12.8
30.0	347.9	-	-50.4	3.23	-	12.8

Member # 8 type: Beam Independent member

width = 18.0 inches f'c = 4.0 ksi Stirrup size = # 4
Depth = 28.0 inches fy = 60.0 ksi

Distance From Left End (ft)	Design Moment Negative (kip-ft)	Design Moment Positive (kip-ft)	Design Shear Force (kip)	Area of steel Bottom (sq.in.)	Area of steel Top (sq.in.)	Stirrup Spacing (inch)
0.0	155.7	-	38.5	1.53	-	12.8
2.1	73.2	-	38.5	1.53	-	12.8
4.3	-	9.9	38.5	-	1.53	12.8
6.4	-	91.6	38.5	-	1.53	12.8
8.6	-	174.1	38.5	-	1.56	12.8
10.7	-	223.9	-7.2	-	2.03	-
12.9	-	208.5	-7.2	-	1.89	-
15.0	-	193.2	-7.2	-	1.74	-
17.1	-	177.8	-7.2	-	1.60	-
19.3	-	162.4	-7.2	-	1.53	-
21.4	-	81.8	-52.8	-	1.53	12.8
23.6	31.4	-	-52.8	1.53	-	12.8
25.7	144.6	-	-52.8	1.53	-	12.8
27.9	257.8	-	-52.8	2.35	-	12.8
30.0	371.0	-	-52.8	3.46	-	12.8

Output File (Contd.)

Design results for structure: Two story space frame - Test Problem 2

Beam design results

Member # 9 type: Beam Independent member

width = 18.0 inches f'c = 4.0 ksi Stirrup size = # 4
Depth = 28.0 inches fy = 60.0 ksi

Distance From Left End (ft)	Design Moment Negative (kip-ft)	Design Moment Positive (kip-ft)	Design Shear Force (kip)	Area of steel Bottom (sq.in.)	Area of steel Top (sq.in.)	Stirrup Spacing (inch)
0.0	347.9	-	50.4	3.23	-	12.8
2.1	240.0	-	50.4	2.18	-	12.8
4.3	132.1	-	50.4	1.53	-	12.8
6.4	24.2	-	50.4	1.53	-	12.8
8.6	-	83.7	50.4	-	1.53	12.8
10.7	-	159.1	4.7	-	1.53	-
12.9	-	169.2	4.7	-	1.53	-
15.0	-	179.3	4.7	-	1.61	-
17.1	-	189.4	4.7	-	1.71	-
19.3	-	199.5	4.7	-	1.80	-
21.4	-	144.4	-40.9	-	1.53	12.8
23.6	-	56.6	-40.9	-	1.53	12.8
25.7	33.5	-	-40.9	1.53	-	12.8
27.9	118.8	-	-40.9	1.53	-	12.8
30.0	206.5	-	-40.9	1.87	-	12.8

Member # 10 type: Beam Independent member

width = 18.0 inches f'c = 4.0 ksi Stirrup size = # 4
Depth = 28.0 inches fy = 60.0 ksi

Distance From Left End (ft)	Design Moment Negative (kip-ft)	Design Moment Positive (kip-ft)	Design Shear Force (kip)	Area of steel Bottom (sq.in.)	Area of steel Top (sq.in.)	Stirrup Spacing (inch)
0.0	371.0	-	52.8	3.46	-	12.8
2.1	257.8	-	52.8	2.35	-	12.8
4.3	144.6	-	52.8	1.53	-	12.8
6.4	31.4	-	52.8	1.53	-	12.8
8.6	-	81.8	52.8	-	1.53	12.8
10.7	-	162.4	7.2	-	1.53	-
12.9	-	177.8	7.2	-	1.60	-
15.0	-	193.2	7.2	-	1.74	-
17.1	-	208.5	7.2	-	1.89	-
19.3	-	223.9	7.2	-	2.03	-
21.4	-	174.1	-38.5	-	1.56	12.8
23.6	-	91.6	-38.5	-	1.53	12.8
25.7	-	9.2	-38.5	-	1.53	12.8
27.9	73.2	-	-38.5	1.53	-	12.8
30.0	155.7	-	-38.5	1.53	-	12.8

Output File (Contd.)

Design results for structure: Two story space frame - Test Problem 2

Beam design results

Member # 11 type: Beam Independent member

width = 12.0 inches f'c = 4.0 ksi Stirrup size = # 4
Depth = 20.0 inches fy = 60.0 ksi

Distance From Left End (ft)	Design Moment Negative (kip-ft)	Design Moment Positive (kip-ft)	Design Shear Force (kip)	Area of steel Bottom (sq.in.)	Area of steel Top (sq.in.)	Stirrup Spacing (inch)
0.0	89.9	-	17.1	1.20	-	8.8
2.1	54.8	-	16.3	0.72	-	8.8
4.3	28.2	-	13.6	0.70	-	8.8
6.4	8.0	-	10.8	0.70	-	8.8
8.6	-	26.2	8.1	-	0.70	-
10.7	-	40.7	5.4	-	0.70	-
12.9	-	49.4	3.2	-	0.70	-
15.0	-	52.3	1.2	-	0.70	-
17.1	-	49.4	-2.7	-	0.70	-
19.3	-	40.7	-5.4	-	0.70	-
21.4	-	27.4	-8.1	-	0.70	-
23.6	-	14.8	-10.8	-	0.70	8.8
25.7	-	4.8	-13.6	-	0.70	8.8
27.9	52.2	-	-16.3	0.70	-	8.8
30.0	89.9	-	-17.1	1.20	-	8.8

Member # 12 type: Beam Independent member

width = 12.0 inches f'c = 4.0 ksi Stirrup size = # 4
Depth = 20.0 inches fy = 60.0 ksi

Distance From Left End (ft)	Design Moment Negative (kip-ft)	Design Moment Positive (kip-ft)	Design Shear Force (kip)	Area of steel Bottom (sq.in.)	Area of steel Top (sq.in.)	Stirrup Spacing (inch)
0.0	82.7	-	17.1	1.10	-	8.8
2.1	44.9	-	16.3	0.70	-	8.8
4.3	17.1	-	13.6	0.70	-	8.8
6.4	0.4	-	10.8	0.70	-	-
8.6	-	33.5	8.1	-	0.70	-
10.7	-	48.0	5.4	-	0.70	-
12.9	-	56.7	2.7	-	0.74	-
15.0	-	59.6	0.7	-	0.78	-
17.1	-	56.7	-2.7	-	0.74	-
19.3	-	48.0	-5.4	-	0.70	-
21.4	-	33.5	-8.1	-	0.70	-
23.6	-	15.6	-10.8	-	0.70	-
25.7	-	1.9	-13.6	-	0.70	8.8
27.9	44.9	-	-16.3	0.70	-	8.8
30.0	82.7	-	-17.1	1.10	-	8.8

Output File (Contd.)

Design results for structure: Two story space frame - Test Problem 2

Beam design results

Member # 13 type: Beam Independent member

width = 16.0 inches f'c = 4.0 ksi Stirrup size = # 4
Depth = 24.0 inches fy = 60.0 ksi

Distance From Left End (ft)	Design Moment Negative (kip-ft)	Design Moment Positive (kip-ft)	Design Shear Force (kip)	Area of steel Bottom (sq.in.)	Area of steel Top (sq.in.)	Stirrup Spacing (inch)
0.0	179.4	-	35.7	1.95	-	10.8
2.1	108.5	-	34.7	1.16	-	10.8
4.3	51.6	-	28.9	1.15	-	10.8
6.4	13.3	-	23.1	1.15	-	10.8
8.6	-	68.6	17.4	-	1.15	10.8
10.7	-	99.6	11.6	-	1.15	-
12.9	-	118.2	7.0	-	1.26	-
15.0	-	124.4	2.7	-	1.33	-
17.1	-	118.2	-5.8	-	1.26	-
19.3	-	99.6	-11.6	-	1.15	-
21.4	-	68.8	-17.4	-	1.15	10.8
23.6	-	42.0	-23.1	-	1.15	10.8
25.7	-	17.1	-28.9	-	1.15	10.8
27.9	98.8	-	-34.7	1.15	-	10.8
30.0	179.4	-	-35.7	1.95	-	10.8

Member # 14 type: Beam Independent member

width = 16.0 inches f'c = 4.0 ksi Stirrup size = # 4
Depth = 24.0 inches fy = 60.0 ksi

Distance From Left End (ft)	Design Moment Negative (kip-ft)	Design Moment Positive (kip-ft)	Design Shear Force (kip)	Area of steel Bottom (sq.in.)	Area of steel Top (sq.in.)	Stirrup Spacing (inch)
0.0	158.0	-	35.7	1.71	-	10.8
2.1	77.4	-	34.7	1.15	-	10.8
4.3	19.6	-	28.9	1.15	-	10.8
6.4	-	46.5	23.1	-	1.15	10.8
8.6	-	89.9	17.4	-	1.15	-
10.7	-	120.9	11.6	-	1.29	-
12.9	-	139.5	5.8	-	1.50	-
15.0	-	145.7	1.2	-	1.57	-
17.1	-	139.5	-5.8	-	1.50	-
19.3	-	120.9	-11.6	-	1.29	-
21.4	-	89.9	-17.4	-	1.15	-
23.6	-	46.5	-23.1	-	1.15	10.8
25.7	-	9.1	-28.9	-	1.15	10.8
27.9	77.4	-	-34.7	1.15	-	10.8
30.0	158.0	-	-35.7	1.71	-	10.8

Output File (Contd.)

Design results for structure: Two story space frame - Test Problem 2

Beam design results

Member # 15 type: Beam Independent member

width = 12.0 inches f'c = 4.0 ksi Stirrup size = # 4
Depth = 20.0 inches fy = 60.0 ksi

Distance From Left End (ft)	Design Moment Negative (kip-ft)	Design Moment Positive (kip-ft)	Design Shear Force (kip)	Area of steel Bottom (sq.in.)	Area of steel Top (sq.in.)	Stirrup Spacing (inch)
0.0	89.9	-	17.1	1.20	-	8.8
2.1	54.8	-	16.3	0.72	-	8.8
4.3	28.2	-	13.6	0.70	-	8.8
6.4	8.0	-	10.8	0.70	-	8.8
8.6	-	26.2	8.1	-	0.70	-
10.7	-	40.7	5.4	-	0.70	-
12.9	-	49.4	3.2	-	0.70	-
15.0	-	52.3	1.2	-	0.70	-
17.1	-	49.4	-2.7	-	0.70	-
19.3	-	40.7	-5.4	-	0.70	-
21.4	-	27.4	-8.1	-	0.70	-
23.6	-	14.8	-10.8	-	0.70	8.8
25.7	-	4.8	-13.6	-	0.70	8.8
27.9	52.2	-	-16.3	0.70	-	8.8
30.0	89.9	-	-17.1	1.20	-	8.8

Member # 16 type: Beam Independent member

width = 12.0 inches f'c = 4.0 ksi Stirrup size = # 4
Depth = 20.0 inches fy = 60.0 ksi

Distance From Left End (ft)	Design Moment Negative (kip-ft)	Design Moment Positive (kip-ft)	Design Shear Force (kip)	Area of steel Bottom (sq.in.)	Area of steel Top (sq.in.)	Stirrup Spacing (inch)
0.0	82.7	-	17.1	1.10	-	8.8
2.1	44.9	-	16.3	0.70	-	8.8
4.3	17.1	-	13.6	0.70	-	8.8
6.4	0.4	-	10.8	0.70	-	-
8.6	-	33.5	8.1	-	0.70	-
10.7	-	48.0	5.4	-	0.70	-
12.9	-	56.7	2.7	-	0.74	-
15.0	-	59.6	0.7	-	0.78	-
17.1	-	56.7	-2.7	-	0.74	-
19.3	-	48.0	-5.4	-	0.70	-
21.4	-	33.5	-8.1	-	0.70	-
23.6	-	15.6	-10.8	-	0.70	-
25.7	-	1.9	-13.6	-	0.70	8.8
27.9	44.9	-	-16.3	0.70	-	8.8
30.0	82.7	-	-17.1	1.10	-	8.8

Output File (Contd.)

Design results for structure: Two story space frame - Test Problem 2

Beam design results

Member # 23 type: Beam Independent member

width = 18.0 inches f'c = 4.0 ksi Stirrup size = # 4
Depth = 28.0 inches fy = 60.0 ksi

Distance From Left End (ft)	Design Moment Negative (kip-ft)	Design Moment Positive (kip-ft)	Design Shear Force (kip)	Area of steel Bottom (sq.in.)	Area of steel Top (sq.in.)	Stirrup Spacing (inch)
0.0	206.5	-	40.9	1.87	-	12.8
2.1	118.8	-	40.9	1.53	-	12.8
4.3	31.1	-	40.9	1.53	-	12.8
6.4	-	56.6	40.9	-	1.53	12.8
8.6	-	144.4	40.9	-	1.53	12.8
10.7	-	199.5	-4.7	-	1.80	-
12.9	-	189.4	-4.7	-	1.71	-
15.0	-	179.3	-4.7	-	1.61	-
17.1	-	169.2	-4.7	-	1.53	-
19.3	-	159.1	-4.7	-	1.53	-
21.4	-	83.7	-50.4	-	1.53	12.8
23.6	24.2	-	-50.4	1.53	-	12.8
25.7	132.1	-	-50.4	1.53	-	12.8
27.9	240.0	-	-50.4	2.18	-	12.8
30.0	347.9	-	-50.4	3.23	-	12.8

Member # 24 type: Beam Independent member

width = 18.0 inches f'c = 4.0 ksi Stirrup size = # 4
Depth = 28.0 inches fy = 60.0 ksi

Distance From Left End (ft)	Design Moment Negative (kip-ft)	Design Moment Positive (kip-ft)	Design Shear Force (kip)	Area of steel Bottom (sq.in.)	Area of steel Top (sq.in.)	Stirrup Spacing (inch)
0.0	155.7	-	38.5	1.53	-	12.8
2.1	73.2	-	38.5	1.53	-	12.8
4.3	-	9.9	38.5	-	1.53	12.8
6.4	-	91.6	38.5	-	1.53	12.8
8.6	-	174.1	38.5	-	1.56	12.8
10.7	-	223.9	-7.2	-	2.03	-
12.9	-	208.5	-7.2	-	1.89	-
15.0	-	193.2	-7.2	-	1.74	-
17.1	-	177.8	-7.2	-	1.60	-
19.3	-	162.4	-7.2	-	1.53	-
21.4	-	81.8	-52.8	-	1.53	12.8
23.6	31.4	-	-52.8	1.53	-	12.8
25.7	144.6	-	-52.8	1.53	-	12.8
27.9	257.8	-	-52.8	2.35	-	12.8
30.0	371.0	-	-52.8	3.46	-	12.8

Output File (Contd.)

Design results for structure: Two story space frame - Test Problem 2

Beam design results

Member # 25 type: Beam Independent member

width = 18.0 inches f'c = 4.0 ksi Stirrup size = # 4
Depth = 28.0 inches fy = 60.0 ksi

Distance From Left End (ft)	Design Moment Negative (kip-ft)	Design Moment Positive (kip-ft)	Design Shear Force (kip)	Area of steel Bottom (sq.in.)	Area of steel Top (sq.in.)	Stirrup Spacing (inch)
0.0	347.9	-	50.4	3.23	-	12.8
2.1	240.0	-	50.4	2.18	-	12.8
4.3	132.1	-	50.4	1.53	-	12.8
6.4	24.2	-	50.4	1.53	-	12.8
8.6	-	83.7	50.4	-	1.53	12.8
10.7	-	159.1	4.7	-	1.53	-
12.9	-	169.2	4.7	-	1.53	-
15.0	-	179.3	4.7	-	1.61	-
17.1	-	189.4	4.7	-	1.71	-
19.3	-	199.5	4.7	-	1.80	-
21.4	-	144.4	-40.9	-	1.53	12.8
23.6	-	56.6	-40.9	-	1.53	12.8
25.7	33.5	-	-40.9	1.53	-	12.8
27.9	118.8	-	-40.9	1.53	-	12.8
30.0	206.5	-	-40.9	1.87	-	12.8

Member # 26 type: Beam Independent member

width = 18.0 inches f'c = 4.0 ksi Stirrup size = # 4
Depth = 28.0 inches fy = 60.0 ksi

Distance From Left End (ft)	Design Moment Negative (kip-ft)	Design Moment Positive (kip-ft)	Design Shear Force (kip)	Area of steel Bottom (sq.in.)	Area of steel Top (sq.in.)	Stirrup Spacing (inch)
0.0	371.0	-	52.8	3.46	-	12.8
2.1	257.8	-	52.8	2.35	-	12.8
4.3	144.6	-	52.8	1.53	-	12.8
6.4	31.4	-	52.8	1.53	-	12.8
8.6	-	81.8	52.8	-	1.53	12.8
10.7	-	162.4	7.2	-	1.53	-
12.9	-	177.8	7.2	-	1.60	-
15.0	-	193.2	7.2	-	1.74	-
17.1	-	208.5	7.2	-	1.89	-
19.3	-	223.9	7.2	-	2.03	-
21.4	-	174.1	-38.5	-	1.56	12.8
23.6	-	91.6	-38.5	-	1.53	12.8
25.7	-	9.2	-38.5	-	1.53	12.8
27.9	73.2	-	-38.5	1.53	-	12.8
30.0	155.7	-	-38.5	1.53	-	12.8

Output File (Contd.)

```
Design results for structure: Two story space frame - Test Problem 2
================================================================

Column design results
----------------------------------------------------------------

Member # 1         type: Column       Reinforcement on four sides
================================================================
width = 18.0 inches       f'c =  4.0 ksi
Depth = 18.0 inches       fy  = 60.0 ksi
----------------------------------------------------------------
APPLIED                                 CALCULATED
Axial load Pu =  54.92 kips             Effective length factors:
Moments:                                kbx =  1.00   kby =  1.00
Smaller end moment M1x =  17.12 k-ft    ksx =  1.00   ksy =  1.00
                   M1y =  28.71 k-ft    Moment magnifiers:
Larger end moment  M2x =  33.79 k-ft    deltaBx = 1.00 deltaBy = 1.00
                   M2y =  39.74 k-ft    deltaSx = 1.00 deltaSy = 1.00
----------------------------------------------------------------
DESIGN RESULTS:
==============
App. load Pu  =  54.92 kips    App. result. mom. Mn  =  63.99 k-ft
All. load Pu1 = 191.03 kips    All. result. mom. Mn1 = 169.53 k-ft

Required steel ratio =  1.00%      Ast Required =  3.24 sq.in.
Use  8 - # 4 and  4 - # 6  bars, Ast =  3.34 sq.in.
Provide # 3 ties at  8.0 inch c/c
----------------------------------------------------------------


Member # 2         type: Column       Reinforcement on four sides
================================================================
width = 18.0 inches       f'c =  4.0 ksi
Depth = 18.0 inches       fy  = 60.0 ksi
----------------------------------------------------------------
APPLIED                                 CALCULATED
Axial load Pu =  57.50 kips             Effective length factors:
Moments:                                kbx =  1.00   kby =  1.00
Smaller end moment M1x = 133.55 k-ft    ksx =  1.00   ksy =  1.00
                   M1y =  64.73 k-ft    Moment magnifiers:
Larger end moment  M2x = 155.68 k-ft    deltaBx = 1.00 deltaBy = 1.00
                   M2y =  88.58 k-ft    deltaSx = 1.00 deltaSy = 1.00
----------------------------------------------------------------
DESIGN RESULTS:
==============
App. load Pu  =  57.50 kips    App. result. mom. Mn  = 220.79 k-ft
All. load Pu1 =  57.55 kips    All. result. mom. Mn1 = 220.84 k-ft

Required steel ratio =  2.04%      Ast Required =  6.61 sq.in.
Use  8 - # 6 and  4 - # 8  bars, Ast =  6.68 sq.in.
Provide # 3 ties at  9.0 inch c/c
----------------------------------------------------------------
```

Output File (Contd.)

```
Design results for structure: Two story space frame - Test Problem 2
=====================================================================

Column design results
---------------------------------------------------------------------

Member # 3         type: Column       Reinforcement on four sides
=====================================================================
width = 18.0 inches       f'c =  4.0 ksi
Depth = 18.0 inches       fy  = 60.0 ksi
---------------------------------------------------------------------
APPLIED                                 CALCULATED
Axial load Pu = 131.97 kips             Effective length factors:
Moments:                                kbx =  1.00   kby =  1.00
Smaller end moment M1x =   0.00 k-ft    ksx =  1.00   ksy =  1.00
                   M1y =  56.93 k-ft    Moment magnifiers:
Larger end moment  M2x =   0.00 k-ft    deltaBx = 1.00 deltaBy = 1.00
                   M2y =  62.37 k-ft    deltaSx = 1.00 deltaSy = 1.00
---------------------------------------------------------------------
DESIGN RESULTS:
==============
App. load Pu  = 131.97 kips    App. result. mom. Mn  =  89.10 k-ft
All. load Pu1 = 397.85 kips    All. result. mom. Mn1 = 213.76 k-ft

Required steel ratio =  1.00%      Ast Required =  3.24 sq.in.
Use  8 - # 4 and  4 - # 6  bars, Ast =  3.34 sq.in.
Provide # 3 ties at  8.0 inch c/c
---------------------------------------------------------------------


Member # 4         type: Column       Reinforcement on four sides
=====================================================================
width = 18.0 inches       f'c =  4.0 ksi
Depth = 18.0 inches       fy  = 60.0 ksi
---------------------------------------------------------------------
APPLIED                                 CALCULATED
Axial load Pu =  66.36 kips             Effective length factors:
Moments:                                kbx =  1.00   kby =  1.00
Smaller end moment M1x =   0.00 k-ft    ksx =  1.00   ksy =  1.00
                   M1y =  52.22 k-ft    Moment magnifiers:
Larger end moment  M2x =   0.00 k-ft    deltaBx = 1.00 deltaBy = 1.00
                   M2y =  75.32 k-ft    deltaSx = 1.00 deltaSy = 1.00
---------------------------------------------------------------------
DESIGN RESULTS:
==============
App. load Pu  =  66.36 kips    App. result. mom. Mn  =  94.44 k-ft
All. load Pu1 = 149.88 kips    All. result. mom. Mn1 = 175.58 k-ft

Required steel ratio =  1.00%      Ast Required =  3.24 sq.in.
Use  8 - # 4 and  4 - # 6  bars, Ast =  3.34 sq.in.
Provide # 3 ties at  8.0 inch c/c
---------------------------------------------------------------------
```

Output File (Contd.)

```
Design results for structure: Two story space frame - Test Problem 2
=====================================================================

Column design results
---------------------------------------------------------------------

Member # 5         type: Column       Reinforcement on four sides
=====================================================================
width = 18.0 inches       f'c =  4.0 ksi
Depth = 18.0 inches       fy  = 60.0 ksi
---------------------------------------------------------------------
APPLIED                                  CALCULATED
Axial load Pu =  54.92 kips              Effective length factors:
Moments:                                 kbx =  1.00   kby =  1.00
Smaller end moment M1x =  17.12 k-ft     ksx =  1.00   ksy =  1.00
                   M1y =  28.71 k-ft     Moment magnifiers:
Larger end moment  M2x =  33.79 k-ft     deltaBx = 1.00 deltaBy = 1.00
                   M2y =  39.74 k-ft     deltaSx = 1.00 deltaSy = 1.00
---------------------------------------------------------------------
DESIGN RESULTS:
==============
App. load Pu  =  54.92 kips    App. result. mom. Mn  =  63.99 k-ft
All. load Pu1 = 191.03 kips    All. result. mom. Mn1 = 169.53 k-ft

Required steel ratio =  1.00%      Ast Required =  3.24 sq.in.
Use  8 - # 4 and  4 - # 6  bars, Ast =  3.34 sq.in.
Provide # 3 ties at  8.0 inch c/c
---------------------------------------------------------------------


Member # 6         type: Column       Reinforcement on four sides
=====================================================================
width = 18.0 inches       f'c =  4.0 ksi
Depth = 18.0 inches       fy  = 60.0 ksi
---------------------------------------------------------------------
APPLIED                                  CALCULATED
Axial load Pu =  57.50 kips              Effective length factors:
Moments:                                 kbx =  1.00   kby =  1.00
Smaller end moment M1x = 133.55 k-ft     ksx =  1.00   ksy =  1.00
                   M1y =  64.73 k-ft     Moment magnifiers:
Larger end moment  M2x = 155.68 k-ft     deltaBx = 1.00 deltaBy = 1.00
                   M2y =  88.58 k-ft     deltaSx = 1.00 deltaSy = 1.00
---------------------------------------------------------------------
DESIGN RESULTS:
==============
App. load Pu  =  57.50 kips    App. result. mom. Mn  = 220.79 k-ft
All. load Pu1 =  57.55 kips    All. result. mom. Mn1 = 220.84 k-ft

Required steel ratio =  2.04%      Ast Required =  6.61 sq.in.
Use  8 - # 6 and  4 - # 8  bars, Ast =  6.68 sq.in.
Provide # 3 ties at  9.0 inch c/c
---------------------------------------------------------------------
```

Output File (Contd.)

```
Design results for structure: Two story space frame - Test Problem 2
=====================================================================

Column design results
---------------------------------------------------------------------

Member # 17         type: Column       Reinforcement on four sides
=====================================================================
width = 18.0 inches       f'c =  4.0 ksi
Depth = 18.0 inches       fy  = 60.0 ksi
---------------------------------------------------------------------
APPLIED                                 CALCULATED
Axial load Pu =  51.02 kips             Effective length factors:
Moments:                                kbx =  1.00   kby =  1.00
Smaller end moment M1x =  17.09 k-ft    ksx =  1.00   ksy =  1.00
                   M1y =   4.35 k-ft    Moment magnifiers:
Larger end moment  M2x =  33.73 k-ft    deltaBx = 1.00 deltaBy = 1.00
                   M2y =  27.53 k-ft    deltaSx = 1.00 deltaSy = 1.00
---------------------------------------------------------------------
DESIGN RESULTS:
==============
App. load Pu  =  51.02 kips    App. result. mom. Mn  =  53.02 k-ft
All. load Pu1 = 223.91 kips    All. result. mom. Mn1 = 167.57 k-ft

Required steel ratio =  1.00%      Ast Required =  3.24 sq.in.
Use  8 - # 4 and  4 - # 6  bars, Ast =  3.34 sq.in.
Provide # 3 ties at  8.0 inch c/c
---------------------------------------------------------------------


Member # 18         type: Column       Reinforcement on four sides
=====================================================================
width = 18.0 inches       f'c =  4.0 ksi
Depth = 18.0 inches       fy  = 60.0 ksi
---------------------------------------------------------------------
APPLIED                                 CALCULATED
Axial load Pu =  57.50 kips             Effective length factors:
Moments:                                kbx =  1.00   kby =  1.00
Smaller end moment M1x = 133.55 k-ft    ksx =  1.00   ksy =  1.00
                   M1y =  64.73 k-ft    Moment magnifiers:
Larger end moment  M2x = 155.68 k-ft    deltaBx = 1.00 deltaBy = 1.00
                   M2y =  88.58 k-ft    deltaSx = 1.00 deltaSy = 1.00
---------------------------------------------------------------------
DESIGN RESULTS:
==============
App. load Pu  =  57.50 kips    App. result. mom. Mn  = 220.79 k-ft
All. load Pu1 =  57.55 kips    All. result. mom. Mn1 = 220.84 k-ft

Required steel ratio =  2.04%      Ast Required =  6.61 sq.in.
Use  8 - # 6 and  4 - # 8  bars, Ast =  6.68 sq.in.
Provide # 3 ties at  9.0 inch c/c
---------------------------------------------------------------------
```

Output File (Contd.)

```
Design results for structure: Two story space frame - Test Problem 2
=====================================================================

Column design results
---------------------------------------------------------------------

Member # 19         type: Column       Reinforcement on four sides
=====================================================================
width = 18.0 inches       f'c =  4.0 ksi
Depth = 18.0 inches       fy  = 60.0 ksi
---------------------------------------------------------------------
APPLIED                                 CALCULATED
Axial load Pu = 124.16 kips             Effective length factors:
Moments:                                kbx =  1.00   kby =  1.00
Smaller end moment M1x =   0.00 k-ft    ksx =  1.00   ksy =  1.00
                   M1y =  13.46 k-ft    Moment magnifiers:
Larger end moment  M2x =   0.00 k-ft    deltaBx = 1.00 deltaBy = 1.00
                   M2y =  40.98 k-ft    deltaSx = 1.00 deltaSy = 1.00
---------------------------------------------------------------------
DESIGN RESULTS:
==============
App. load Pu  = 124.16 kips    App. result. mom. Mn  =  57.85 k-ft
All. load Pu1 = 526.82 kips    All. result. mom. Mn1 = 209.44 k-ft

Required steel ratio =  1.00%      Ast Required =  3.24 sq.in.
Use  8 - # 4 and  4 - # 6  bars, Ast =  3.34 sq.in.
Provide # 3 ties at  8.0 inch c/c
---------------------------------------------------------------------


Member # 20         type: Column       Reinforcement on four sides
=====================================================================
width = 18.0 inches       f'c =  4.0 ksi
Depth = 18.0 inches       fy  = 60.0 ksi
---------------------------------------------------------------------
APPLIED                                 CALCULATED
Axial load Pu =  63.99 kips             Effective length factors:
Moments:                                kbx =  1.00   kby =  1.00
Smaller end moment M1x =   0.00 k-ft    ksx =  1.00   ksy =  1.00
                   M1y =  41.42 k-ft    Moment magnifiers:
Larger end moment  M2x =   0.00 k-ft    deltaBx = 1.00 deltaBy = 1.00
                   M2y =  42.46 k-ft    deltaSx = 1.00 deltaSy = 1.00
---------------------------------------------------------------------
DESIGN RESULTS:
==============
App. load Pu  =  63.99 kips    App. result. mom. Mn  =  52.99 k-ft
All. load Pu1 = 322.89 kips    All. result. mom. Mn1 = 174.33 k-ft

Required steel ratio =  1.00%      Ast Required =  3.24 sq.in.
Use  8 - # 4 and  4 - # 6  bars, Ast =  3.34 sq.in.
Provide # 3 ties at  8.0 inch c/c
---------------------------------------------------------------------
```

Output File (Contd.)

```
Design results for structure: Two story space frame - Test Problem 2
=====================================================================

Column design results
---------------------------------------------------------------------

Member # 21         type: Column       Reinforcement on four sides
=====================================================================
width = 18.0 inches       f'c =  4.0 ksi
Depth = 18.0 inches       fy  = 60.0 ksi
---------------------------------------------------------------------
APPLIED                                  CALCULATED
Axial load Pu =  51.02 kips              Effective length factors:
Moments:                                 kbx =  1.00   kby =  1.00
Smaller end moment M1x =  17.09 k-ft     ksx =  1.00   ksy =  1.00
                   M1y =   4.35 k-ft     Moment magnifiers:
Larger end moment  M2x =  33.73 k-ft     deltaBx = 1.00 deltaBy = 1.00
                   M2y =  27.53 k-ft     deltaSx = 1.00 deltaSy = 1.00
---------------------------------------------------------------------
DESIGN RESULTS:
==============
App. load Pu  =  51.02 kips    App. result. mom. Mn  =  53.02 k-ft
All. load Pu1 = 223.91 kips    All. result. mom. Mn1 = 167.57 k-ft

Required steel ratio =  1.00%      Ast Required =  3.24 sq.in.
Use  8 - # 4 and  4 - # 6  bars, Ast =  3.34 sq.in.
Provide # 3 ties at  8.0 inch c/c
---------------------------------------------------------------------


Member # 22         type: Column       Reinforcement on four sides
=====================================================================
width = 18.0 inches       f'c =  4.0 ksi
Depth = 18.0 inches       fy  = 60.0 ksi
---------------------------------------------------------------------
APPLIED                                  CALCULATED
Axial load Pu =  57.50 kips              Effective length factors:
Moments:                                 kbx =  1.00   kby =  1.00
Smaller end moment M1x = 133.55 k-ft     ksx =  1.00   ksy =  1.00
                   M1y =  64.73 k-ft     Moment magnifiers:
Larger end moment  M2x = 155.68 k-ft     deltaBx = 1.00 deltaBy = 1.00
                   M2y =  88.58 k-ft     deltaSx = 1.00 deltaSy = 1.00
---------------------------------------------------------------------
DESIGN RESULTS:
==============
App. load Pu  =  57.50 kips    App. result. mom. Mn  = 220.79 k-ft
All. load Pu1 =  57.55 kips    All. result. mom. Mn1 = 220.84 k-ft

Required steel ratio =  2.04%      Ast Required =  6.61 sq.in.
Use  8 - # 6 and  4 - # 8  bars, Ast =  6.68 sq.in.
Provide # 3 ties at  9.0 inch c/c
---------------------------------------------------------------------
```

Vita

Jayendra R. Patel was born in Baroda, India on August 18, 1967. In August 1988, he graduated from B. & B. Polytechnic, Gujarat, India with a Diploma in Civil Engineering. In the final examination for the Diploma, he placed first among approximately 1600 students. In August 1991, he was awarded the Bachelor of Civil Engineering degree by Gujarat University. In August 1992, he enrolled in the Structures Division in Civil Engineering Department at Virginia Polytechnic Institute and State University where he earned the Master of Science in Civil Engineering degree in May 1994.

Jayendra R Patel.