CAMBRIDGE INSTITUTE OF TECHNOLOGY

K.R. PURAM, BANGALORE – 560 036, Ph: 080-2561 8798 / 2561 8799
Fax: 080-2561 8789, email: principal@cambridge.edu.in

Affiliated to VTU, Belagavi|Approved by AICTE, New Delhi|NAAC& NBA Accredited|
UGC 2(f) Certified| Recognized by Govt. of Karnataka

Department of Artificial Intelligence and Machine Learning

Academic year : 2023-2024

LABORATORY MANUAL

Semester : 3rd

Sub : DATA STRUCTURES LABORATORY

Sub code : BCSL305

Student Name: _____

USN : _____

Section : _____

Batch : _____

Prepared by : Prof. R.Geetha

PROGRAM OUTCOMES

Engineering graduates will be able to

PO1 : Engineering Knowledge: Apply the knowledge of mathematics, science, engineering fundamentals and an engineering specialization to the solution of complex engineering problems.

PO2 : Problem Analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences and Engineering sciences.

PO3 : Design/Development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

PO4 : Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO5 : Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO6 : The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO7 : Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8 : Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9 : Individual and team work: Function effectively as an individual and as a member or leader in diverse teams, and in multidisciplinary settings.

PO10 : Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO11 : Project management and finance: Demonstrate knowledge and understanding of the engineering management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO12 : Life-long learning: Recognize the need for and have the preparation and ability to engage in independent and lifelong learning in the broadest context of technological change.

Course Learning Objectives

This course (BCSL305) will enable students to:

- To make the student learn a programming language.
- To learn problem solving techniques.
- To teach the student to write programs in C and to solve the problems.

Course Outcomes

The student should be able to:

CO1: Analyze various linear and non-linear data structures

CO2: Demonstrate the working nature of different types of data structures and their applications

CO3: Use appropriate searching and sorting algorithms for the give scenario.

CO4: Apply the appropriate data structure for solving real world problems

DATA STRUCTURES LABORATORY			
Course Code	**BCSL305**	**CIE Marks**	50
Number of Contact Hours/Week	0:0:2	**SEE Marks**	50
Total Number of Lab Contact Hours	28	**Exam Hours**	03
Credits – 1			

Descriptions (if any):

- Implement all the programs in "C " Programming Language and Linux OS.

Programs List:

1.	Develop a Program in C for the following: a) Declare a calendar as an array of 7 elements (A dynamically Created array) to represent 7 days of a week. Each Element of the array is a structure having three fields. The first field is the name of the Day (A dynamically allocated String), The second field is the date of the Day (A integer), the third field is the description of the activity for a particular day (A dynamically allocated String). b) Write functions create(), read() and display(); to create the calendar, to read the data from the keyboard and to print weeks activity details report on screen.
2.	Develop a Program in C for the following operations on Strings. a. Read a main String (STR), a Pattern String (PAT) and a Replace String (REP) b. Perform Pattern Matching Operation: Find and Replace all occurrences of PAT in STR with REP if PAT exists in STR. Report suitable messages in case PAT does not exist in STR Support the program with functions for each of the above operations. Don't use Built-in functions.
3.	Develop a menu driven Program in C for the following operations on STACK of Integers(Array Implementation of Stack with maximum size MAX) a. Push an Element on to Stack b. Pop an Element from Stack c. Demonstrate how Stack can be used to check Palindrome d. Demonstrate Overflow and Underflow situations on Stack e. Display the status of Stack f. Exit Support the program with appropriate functions for each of the above operations
4.	Develop a Program in C for converting an Infix Expression to Postfix Expression. Program should support for both parenthesized and free parenthesized expressions with the operators: +, -, *, /, % (Remainder), ^ (Power) and alphanumeric operands.
5.	Develop a Program in C for the following Stack Applications a. Evaluation of Suffix expression with single digit operands and operators: +, -, *, /, %, ^ b. Solving Tower of Hanoi problem with n disks
6.	Develop a menu driven Program in C for the following operations on Circular QUEUE of Characters (Array Implementation of Queue with maximum size MAX) a. Insert an Element on to Circular QUEUE b. Delete an Element from Circular QUEUE c. Demonstrate Overflow and Underflow situations on Circular QUEUE d. Display the status of Circular QUEUE e. Exit Support the program with appropriate functions for each of the above operations
7.	Develop a menu driven Program in C for the following operations on Singly Linked List (SLL) of Student Data with the fields: USN, Name, Programme, Sem, PhNo a. Create a SLL of N Students Data by using front insertion. b. Display the status of SLL and count the number of nodes in it c. Perform Insertion / Deletion at End of SLL d. Perform Insertion / Deletion at Front of SLL(Demonstration of stack) e. Exit

8.	Develop a menu driven Program in C for the following operations on Doubly Linked List (DLL) of Employee Data with the fields: SSN, Name, Dept, Designation, Sal, PhNo a. Create a DLL of N Employees Data by using end insertion. b. Display the status of DLL and count the number of nodes in it c. Perform Insertion and Deletion at End of DLL d. Perform Insertion and Deletion at Front of DLL e. Demonstrate how this DLL can be used as Double Ended Queue. f. Exit
9.	Develop a Program in C for the following operationson Singly Circular Linked List (SCLL) with header nodes a. Represent and Evaluate a Polynomial P(x,y,z) = 6x2y2z-4yz5+3x3yz+2xy5z-2xyz3 b. Find the sum of two polynomials POLY1(x,y,z) and POLY2(x,y,z) and store the result in POLYSUM(x,y,z) Support the program with appropriate functions for each of the above operations
10.	Develop a menu driven Program in C for the following operations on Binary Search Tree (BST) of Integers . a. Create a BST of N Integers: 6, 9, 5, 2, 8, 15, 24, 14, 7, 8, 5, 2 b. Traverse the BST in Inorder, Preorder and Post Order c. Search the BST for a given element (KEY) and report the appropriate message d. Exit
11.	Develop a Program in C for the following operations on Graph(G) of Cities a. Create a Graph of N cities using Adjacency Matrix. b. Print all the nodes reachable from a given starting node in a digraph using DFS/BFS method
12.	Given a File of N employee records with a set K of Keys (4-digit) which uniquely determine the records in file F. Assume that file F is maintained in memory by a Hash Table (HT) of m memory locations with L as the set of memory addresses (2-digit) of locations in HT. Let the keys in K and addresses in L are Integers. Develop a Program in C that uses Hash function H: K →L as H(K)=K mod m (remainder method), and implement hashing technique to map a given key K to the address space L. Resolve the collision (if any) using linear probing.

****

1. Develop a Program in C for the following:
   a) Declare a calendar as an array of 7 elements (A dynamically Created array) to represent 7 days of a week. Each Element of the array is a structure having three fields. The first field is the name of the Day (A dynamically allocated String), The second field is the date of the Day (A integer), the third field is the description of the activity for a particular day (A dynamically allocated String).

   b) Write functions create(), read() and display(); to create the calendar, to read the data from the keyboard and to print weeks activity details report on screen.

****

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#define NUM_DAYS_IN_WEEK 7

// Structure to represent a day
typedef struct
{
   char *name;    // Dynamically allocated string for the day name
   int date;        // Date of the day
   char *activity ; // Dynamically allocated string for the activity description
}daytype;
int i;

void fnFreeCal(daytype *);
void fnDispCal(daytype *);
void fnReadCal(daytype *);
daytype *fnCreateCal();

int main()
{
   // Create the calendar
   daytype *weeklyCalendar = fnCreateCal();

   // Read data from the keyboard
   fnReadCal(weeklyCalendar);

   // Display the week's activity details
   fnDispCal(weeklyCalendar);

   // Free allocated memory
   fnFreeCal(weeklyCalendar);

   return 0;
}
```

```
daytype *fnCreateCal()
{
  daytype *calendar = (daytype *)malloc(NUM_DAYS_IN_WEEK * sizeof(daytype));
   for(i = 0; i < NUM_DAYS_IN_WEEK; i++)
  {
      calendar[i].name= NULL;
      calendar[i].date = 0;
      calendar[i].activity= NULL;
   }
   return calendar;
}

void fnReadCal(daytype *calendar)
{
  char cChoice;

   for(i=0; i< NUM_DAYS_IN_WEEK; i++)
  {
      printf("Do you want to enter details for day %d [Y/N]: ", i + 1);
      scanf("%c", &cChoice);
              getchar();

      if(tolower(cChoice) =='n')
           continue;
      printf("Day Name: ");
      char nameBuffer[50];
      scanf("%s",nameBuffer);
      calendar[i].name = strdup(nameBuffer);  // Dynamically allocate and copy the string
      printf("Date: ");
      scanf("%d", &calendar[i].date);
      printf("Activity: ");
      char activityBuffer[100];
      scanf("%s",activityBuffer);  // Read the entire line, including spaces
      calendar[i].activity= strdup(activityBuffer);
      printf("\n");
      getchar();              //remove trailing enter character in input buffer
   }
}

void fnDispCal(daytype *calendar)
{
   printf("\n nWeek's Activity Details:n");
   for(i = 0; i < NUM_DAYS_IN_WEEK; i++)
  {
      printf("Day %d\n", i + 1);
           if(calendar[i].date == 0)
           {
                    printf("No Activityn\n");
                    continue;
           }

      printf("  Day Name: %s\n", calendar[i].name);
```

```
            printf("  Date: %d\n", calendar[i].date);
            printf("  Activity: %s\n", calendar[i].activity);
          }
       }

       void fnFreeCal(daytype *calendar)
       {
          for(i = 0; i < NUM_DAYS_IN_WEEK; i++)
         {
            free(calendar[i].name);
     // free(calendar[i].date);
      free(calendar[i].activity);
   }
   free(calendar);
}
```

```
*********OUTPUT**********************
Do you want to enter details for day 1 [Y/N]: N
Do you want to enter details for day 2 [Y/N]: Y
Day Name: Monday
Date: 10
Activity: Meeting with Chairman.

Do you want to enter details for day 3 [Y/N]: N
Do you want to enter details for day 4 [Y/N]: N
Do you want to enter details for day 5 [Y/N]: Y
Day Name: Thursday
Date: 13
Activity: Product Survey

Do you want to enter details for day 6 [Y/N]: Y
Day Name: Friday
Date: 14
Activity: Budget Breakdown and Planning

Do you want to enter details for day 7 [Y/N]: Y
Day Name: Saturday
Date: 15
Activity: Outing with family


Week's Activity Details:
Day 1:
No Activity

Day 2:
  Day Name: Monday
  Date: 10
  Activity: Meeting with Chairman.

Day 3:
No Activity
```

Day 4:
No Activity

Day 5:
Day Name: Thursday
Date: 13
Activity: Product Survey

Day 6:
Day Name: Friday
Date: 14
Activity: Budget Breakdown and Planning

Day 7:
Day Name: Saturday
Date: 15
Activity: Outing with family

****

2. Develop a Program in C for the following operations on Strings.
   a. Read a main String (STR), a Pattern String (PAT) and a Replace String (REP)
   b. Perform Pattern Matching Operation: Find and Replace all occurrences of PAT in STR with REP if PAT exists in STR. Report suitable messages in case PAT does not exist in STR

Support the program with functions for each of the above operations. Don't use Built-in functions.

****

Program

```
#include<stdio.h>
main()
{
        char str[100],pat[100],rep[100],ans[100];
        int i,j,c,m,k;

        printf("\nEnter a string \n");
        gets(str);
        printf("\nEnter a search string \n");

        gets(pat);
        printf("\nEnter a replace string \n");

        gets(rep);
        i = m = c = j = 0;
        while ( str[c] != '\0')
        {
                if ( str[m] == pat[i] )
                {
                        i++;
                        m++;
                if ( pat[i] == '\0')
                {

                for(k=0; rep[k] != '\0';k++,j++)
                        ans[j] = rep[k];
                        i=0;
                        c=m;
                }
                }
                else
                {
                        ans[j] = str[c];
                        j++;
                        c++;
                        m = c;
                        i=0;
                }
        }
        ans[j] = '\0';
        printf("\nThe resultant string is\n%s" ,ans);
}
```

Sample Output:

Enter a string
Raju and Ramu went to shop to buy milk and cakes.

Enter a search string
and

Enter a replace string
&

The resultant string is
Raju & Ramu went to shop to buy milk & cakes.

****

3. Design, Develop and Implement a menu driven Program in C for the following operations on STACK of Integers (Array Implementation of Stack with maximum size MAX)

a. Push an Element on to Stack

b. Pop an Element from Stack

c. Demonstrate how Stack can be used to check Palindrome

d. Demonstrate Overflow and Underflow situations on Stack

e. Display the status of Stack

f. Exit

Support the program with appropriate functions for each of the above operations

****

```
#include<stdio.h>
#include<stdlib.h>
#define MAX 5

int s[MAX];
int top = -1;

void push(int item);
int pop();
void palindrome();
void display();

void main()
{
        int choice, item;
        while(1)
        {
                printf("\n\n\n\n~~~~~~Menu~~~~~~ : ");
                printf("\n=>1.Push an Element to Stack and Overflow demo ");
                printf("\n=>2.Pop an Element from Stack and Underflow demo");
                printf("\n=>3.Palindrome demo ");
                printf("\n=>4.Display ");
                printf("\n=>5.Exit");
                printf("\nEnter your choice: ");
                scanf("%d", &choice);
                switch(choice)
                {
                        case 1:         printf("\nEnter an element to be pushed: ");
                                        scanf("%d", &item);
                                        push(item);
                                        break;
                        case 2:         item = pop();
                                        if(item != -1)
                                                printf("\nElement popped is: %d", item);
```

```
                                        break;
                        case 3:         palindrome();
                                        break;
                        case 4:         display();
                                        break;
                        case 5:         exit(1);
                        default:        printf("\nPlease enter valid choice ") ;
                                        break;
                }
        }
}

void push(int item)
{
        if(top == MAX-1)
        {
                printf("\n~~~~Stack overflow~~~~");
                return;
        }

        top = top + 1 ;
        s[top] = item;
}

int pop()
{
        int item;
        if(top == -1)
        {
                printf("\n~~~~Stack underflow~~~~");
                return -1;
        }
        item = s[top];
        top = top - 1;
        return item;
}

void display()
{
        int i;
        if(top == -1)
        {
                printf("\n~~~~Stack is empty~~~~");
                return;
        }
        printf("\nStack elements are:\n ");
        for(i=top; i>=0 ; i--)
                printf("| %d |\n", s[i]);
}

void palindrome()
{
```

```
        int flag=1,i;
        printf("\nStack content are:\n");
        for(i=top; i>=0 ; i--)
                printf("| %d |\n", s[i]);

        printf("\nReverse of stack content are:\n");
        for(i=0; i<=top; i++)
                printf("| %d |\n", s[i]);

        for(i=0; i<=top/2; i++)
        {
                if( s[i] != s[top-i] )
                {
                        flag = 0;
                        break;
                }
        }
        if(flag == 1)
        {
                printf("\nIt is palindrome number");
        }
        else
        {
                printf("\nIt is not a palindrome number");
        }
}
```

********Output*************

~~~~~~Menu~~~~~~ :
=>1.Push an Element to Stack and Overflow demo
=>2.Pop an Element from Stack and Underflow demo
=>3.Palindrome demo
=>4.Display
=>5.Exit
Enter your choice: **1**
Enter an element to be pushed: 11

~~~~~~Menu~~~~~~ :
=>1.Push an Element to Stack and Overflow demo
=>2.Pop an Element from Stack and Underflow demo
=>3.Palindrome demo
=>4.Display
=>5.Exit
Enter your choice: **1**
Enter an element to be pushed: 12

~~~~~~Menu~~~~~~ :
=>1.Push an Element to Stack and Overflow demo
=>2.Pop an Element from Stack and Underflow demo
=>3.Palindrome demo
=>4.Display
=>5.Exit
Enter your choice: **1**
~~~~~~

Enter an element to be pushed: 13

~~~~~~Menu~~~~~~ :
=>1.Push an Element to Stack and Overflow demo
=>2.Pop an Element from Stack and Underflow demo
=>3.Palindrome demo
=>4.Display
=>5.Exit
Enter your choice: **1**
Enter an element to be pushed: 14

~~~~~~Menu~~~~~
=>1.Push an Element to Stack and Overflow demo
=>2.Pop an Element from Stack and Underflow demo
=>3.Palindrome demo
=>4.Display
=>5.Exit
Enter your choice: **1**
Enter an element to be pushed: 15

~~~~~~Menu~~~~~
=>1.Push an Element to Stack and Overflow demo
=>2.Pop an Element from Stack and Underflow demo
=>3.Palindrome demo
=>4.Display
=>5.Exit
Enter your choice: **1**
Enter an element to be pushed: 16
~~~~Stack overflow~~~~

~~~~~~Menu~~~~~
=>1.Push an Element to Stack and Overflow demo
=>2.Pop an Element from Stack and Underflow demo
=>3.Palindrome demo
=>4.Display
=>5.Exit
Enter your choice: **4**

Stack elements are:
| 15 |
| 14 |
| 13 |
| 12 |
| 11 |

~~~~~~Menu~~~~~
=>1.Push an Element to Stack and Overflow demo
=>2.Pop an Element from Stack and Underflow demo
=>3.Palindrome demo
=>4.Display
=>5.Exit
Enter your choice: **2**
Element popped is: 15

~~~~~~Menu~~~~~
=>1.Push an Element to Stack and Overflow demo
=>2.Pop an Element from Stack and Underflow demo

=>3.Palindrome demo
=>4.Display
=>5.Exit
Enter your choice: **4**
Stack elements are:
| 14 |
| 13 |
| 12 |
| 11 |
~~~~~~Menu~~~~~
=>1.Push an Element to Stack and Overflow demo
=>2.Pop an Element from Stack and Underflow demo
=>3.Palindrome demo
=>4.Display
=>5.Exit
Enter your choice: **2**
Element popped is: 14

~~~~~~Menu~~~~~
=>1.Push an Element to Stack and Overflow demo
=>2.Pop an Element from Stack and Underflow demo
=>3.Palindrome demo
=>4.Display
=>5.Exit
Enter your choice: **2**
Element popped is: 13

~~~~~~Menu~~~~~
=>1.Push an Element to Stack and Overflow demo
=>2.Pop an Element from Stack and Underflow demo
=>3.Palindrome demo
=>4.Display
=>5.Exit
Enter your choice: **2**
Element popped is: 12

~~~~~~Menu~~~~~
=>1.Push an Element to Stack and Overflow demo
=>2.Pop an Element from Stack and Underflow demo
=>3.Palindrome demo
=>4.Display
=>5.Exit
Enter your choice: **2**
Element popped is: 11

~~~~~~Menu~~~~~
=>1.Push an Element to Stack and Overflow demo
=>2.Pop an Element from Stack and Underflow demo
=>3.Palindrome demo
=>4.Display
=>5.Exit
Enter your choice: **2**
~~~~Stack underflow~~~~

~~~~~~Menu~~~~~
=>1.Push an Element to Stack and Overflow demo
=>2.Pop an Element from Stack and Underflow demo

=>3.Palindrome demo
=>4.Display
=>5.Exit
Enter your choice: **4**
~~~~Stack is empty~~~~

~~~~~~Menu~~~~~
=>1.Push an Element to Stack and Overflow demo
=>2.Pop an Element from Stack and Underflow demo
=>3.Palindrome demo
=>4.Display
=>5.Exit
Enter your choice: **1**
Enter an element to be pushed: 11

~~~~~~Menu~~~~~
=>1.Push an Element to Stack and Overflow demo
=>2.Pop an Element from Stack and Underflow demo
=>3.Palindrome demo
=>4.Display
=>5.Exit
Enter your choice: **1**
Enter an element to be pushed: 22

~~~~~~Menu~~~~~
=>1.Push an Element to Stack and Overflow demo
=>2.Pop an Element from Stack and Underflow demo
=>3.Palindrome demo
=>4.Display
=>5.Exit
Enter your choice: **1**
Enter an element to be pushed: 11

~~~~~~Menu~~~~~
=>1.Push an Element to Stack and Overflow demo
=>2.Pop an Element from Stack and Underflow demo
=>3.Palindrome demo
=>4.Display
=>5.Exit
Enter your choice: **3**
Stack content are:
| 11 |
| 22 |
| 11 |

Reverse of stack content are:
| 11 |
| 22 |
| 11 |

It is palindrome number

~~~~~~Menu~~~~~
=>1.Push an Element to Stack and Overflow demo
=>2.Pop an Element from Stack and Underflow demo
=>3.Palindrome demo
=>4.Display
~~~~~~

=>5.Exit
Enter your choice: **2**
Element popped is: 11

~~~~~~Menu~~~~~
=>1.Push an Element to Stack and Overflow demo
=>2.Pop an Element from Stack and Underflow demo
=>3.Palindrome demo
=>4.Display
=>5.Exit
Enter your choice: **2**
Element popped is: 22

~~~~~~Menu~~~~~
=>1.Push an Element to Stack and Overflow demo
=>2.Pop an Element from Stack and Underflow demo
=>3.Palindrome demo
=>4.Display
=>5.Exit
Enter your choice: **2**
Element popped is: 11

~~~~~~Menu~~~~~
=>1.Push an Element to Stack and Overflow demo
=>2.Pop an Element from Stack and Underflow demo
=>3.Palindrome demo
=>4.Display
=>5.Exit
Enter your choice: **1**
Enter an element to be pushed: 11

~~~~~~Menu~~~~~
=>1.Push an Element to Stack and Overflow demo
=>2.Pop an Element from Stack and Underflow demo
=>3.Palindrome demo
=>4.Display
=>5.Exit
Enter your choice: **1**
Enter an element to be pushed: 22

~~~~~~Menu~~~~~
=>1.Push an Element to Stack and Overflow demo
=>2.Pop an Element from Stack and Underflow demo
=>3.Palindrome demo
=>4.Display
=>5.Exit
Enter your choice: **1**
Enter an element to be pushed: 33

~~~~~~Menu~~~~~
=>1.Push an Element to Stack and Overflow demo
=>2.Pop an Element from Stack and Underflow demo
=>3.Palindrome demo
=>4.Display
=>5.Exit
Enter your choice: **3**
Stack content are:

| 33 |
| 22 |
| 11 |

Reverse of stack content are:
| 11 |
| 22 |
| 33 |

It is not a palindrome number

~~~~~~Menu~~~~~
=>1.Push an Element to Stack and Overflow demo
=>2.Pop an Element from Stack and Underflow demo
=>3.Palindrome demo
=>4.Display
~~~~~~

****

4. Design, Develop and Implement a Program in C for converting an Infix Expression to Postfix Expression. Program should support for both parenthesized and free parenthesized expressions with the operators: +, -, *, /, %(Remainder), ^(Power) and alphanumeric operands.

****

```
#include<stdio.h>
#include<stdlib.h>

void evaluate();
void push(char);
char pop();
int prec(char);

char infix[30], postfix[30], stack[30];
int top = -1;

void main()
{
        printf("\nEnter the valid infix expression:\t");
        scanf("%s", infix);
        evaluate();
        printf("\nThe entered infix expression is :\n %s \n", infix);
        printf("\nThe corresponding postfix expression is :\n %s \n", postfix);
}

void evaluate()
{
        int i = 0, j = 0;
        char symb, temp;

        push('#');

        for(i=0; infix[i] != '\0'; i++)
        {
                symb = infix[i];
                switch(symb)
                {
                        case '(' :      push(symb);
                                        break;

                        case ')' :      temp = pop();
                                        while(temp != '(' )
                                        {
                                                postfix[j] = temp;
                                                j++;
                                                temp = pop();
                                        }
                                        break;
```

```
                        case '+' :
                        case '-' :
                        case '*' :
                        case '/' :
                        case '%' :
                        case '^' :
                        case '$'  :     while( prec(stack[top]) >= prec(symb) )
                                        {
                                                temp = pop();
                                                postfix[j] = temp;
                                                j++;
                                        }
                                        push(symb);
                                        break;
                        default:        postfix[j] = symb;
                                        j++;
                }
        }
        while(top > 0)
        {
                temp = pop();
                postfix[j] = temp;
                j++;
        }
        postfix[j] = '\0';
}

 void push(char item)
{
        top = top+1;
        stack[top] = item;
}

 char pop()
{
        char item;
        item = stack[top];
        top = top-1;
        return item;
}

int prec(char symb)
{
        int p;
        switch(symb)
        {
                case '#' :      p = -1;
                                break;

                case '(' :
                case ')' :      p = 0;
                                break;
```

```
                case '+' :
                case '-' :          p = 1;
                                    break;

                case '*' :
                case '/' :
                case '%' :          p = 2;
                                    break;

                case '^' :
                case '$' :          p = 3;
                                    break;
        }
        return p;
}
```

Output:

Enter the valid infix expression: **(a+b)+c/d*e**

The entered infix expression is :
(a+b)+c/d*e

The corresponding postfix expression is :
ab+cd/e*+

****

5. Design, Develop and Implement a Program in C for the following Stack Applications

a. Evaluation of Suffix expression with single digit operands and operators: +, -, *, /, %, ^.

****

```
 #include<stdio.h>
#include<stdlib.h>
#include<math.h>

int i, top = -1;
int op1, op2, res, s[20];
char postfix[90], symb;

void push(int item)
{
        top = top+1;
        s[top] = item;
}

int pop()
{
        int item;
        item  =  s[top];
        top = top-1;
        return item;
}

void main()
{
        printf("\nEnter a valid postfix expression:\n");
        scanf("%s", postfix);
        for(i=0; postfix[i]!='\0'; i++)
        {
                symb = postfix[i];
                if(isdigit(symb))
                {
                        push(symb - '0');
                }
                else
                {
                        op2 = pop();
                        op1 = pop();
                        switch(symb)
                        {
                                case '+':       push(op1+op2);
                                                break;
                                case '-':       push(op1-op2);
                                                break;
                                case '*':       push(op1*op2);
                                                break;
                                case '/':       push(op1/op2);
```

```
                                        break;
                                case '%':       push(op1%op2);
                                        break;
                                case '$':
                                case '^':       push(pow(op1, op2));
                                        break;
                                default :   push(0);
                        }
                }
        }
        res = pop();
        printf("\n Result = %d", res);
}
```

Output:

To compile in Linux: gcc –lm 5.c

Enter a valid postfix expression:
623+-382/+*2$3+
Result = 52

Enter a valid postfix expression:
42$3*3-84/11+/+
Result = 46

b: Solving Tower of Hanoi problem with n disks

```
#include<stdio.h>
void towers(int, char, char, char);
void main()
{int num;
printf("ENTER THE NUMBER OF DISKS\n");
scanf("%d",&num);
printf("THE SEQUENCE INVOLVED ARE:\n");
towers(num,'A','C','B');
}
void towers(int num, char frompeg, char topeg, char auxpeg)
{
if(num==1)
{printf("MOVE DISK 1 FROM %c TO %c\n",frompeg,topeg);
return;
}
towers(num-1,frompeg,auxpeg,topeg);
printf("MOVE DISK %d FROM %c TO %c\n",num,frompeg,topeg);
towers(num-1,auxpeg,topeg,frompeg);
}
```

Output:

```
ENTER THE NUMBER OF DISKS
3
THE SEQUENCE INVOLVED ARE:
MOVE DISK 1 FROM  A  TO C
MOVE DISK 2 FROM A  TO B
MOVE DISK 1 FROM  C  TO B
MOVE DISK 3 FROM  A  TO C
MOVE DISK 2 FROM  B  TO C
MOVE DISK 1 FROM  A  TO C
```

****

6. Design, Develop and Implement a menu driven Program in C for the following operations on Circular QUEUE of Characters (Array Implementation of Queue with maximum size MAX)

a. Insert an Element on to Circular QUEUE

b. Delete an Element from Circular QUEUE

c. Demonstrate Overflow and Underflow situations on Circular QUEUE

d. Display the status of Circular QUEUE

e. Exit

Support the program with appropriate functions for each of the above operations

****

```
#include <stdio.h>
#include<stdlib.h>
#include<stdio_ext.h>
#define MAX 3
char cq[MAX];
int front = -1, rear = -1;

void insert(char);
void delete();
void display();
void main()
{
        int ch;
        char item;
        while(1)
        {
                printf("\n\n~~Main Menu~~");
                printf("\n==> 1. Insertion and Overflow Demo");
                printf("\n==> 2. Deletion and Underflow Demo");
                printf("\n==> 3. Display");
                printf("\n==> 4. Exit");
                printf("\nEnter Your Choice: ");
                scanf("%d", &ch);
                __fpurge(stdin);
                switch(ch)
                {
                        case 1:     printf("\n\nEnter the element to be inserted: ");
                                    scanf("%c", &item);
                                    insert(item);
                                    break;
                        case 2:     delete();
                                    break;
                        case 3:     display();
                                    break;
                         case 4:    exit(0);
                         default:   printf("\n\nPlease enter a valid choice");
                }
```

```
            }
}

void insert(char item)
{
            if(front == (rear+1)%MAX)
            {
                        printf("\n\n~~Circular Queue Overflow~~");
            }
            else
            {
                        if(front == -1)
                                    front = rear = 0;
                        else
                                    rear = (rear+1)%MAX;
                        cq[rear] = item;
             }
}

void delete()
{
            char item;
            if(front == -1)
            {
                        printf("\n\n~~Circular Queue Underflow~~");
            }
            else
            {
                        item = cq[front];
                        printf("\n\nDeleted element from the queue is: %c ",item );

                        if(front == rear) //only one element
                                    front = rear = -1;
                        else
                                    front = (front+1)%MAX;
             }
}


void display ()
{
              int i ;
              if(front == -1)
              {
                          printf("\n\nCircular Queue Empty");
              }
              else
              {
                          printf("\nCircular Queue contents are:\n");
                          printf("Front[%d]-> ", front);
                          for(i = front; i != rear ; i = (i+1)%MAX)
                          {
```

```
                        printf(" %c", cq[i]);
                }
                printf(" %c", cq[i]);
                printf(" <-[%d]Rear", rear);
        }
}
```

Output:

~~Main Menu~~
==> 1. Insertion and Overflow Demo
==> 2. Deletion and Underflow Demo
==> 3. Display
==> 4. Exit
Enter Your Choice: **1**

Enter the element to be inserted: A

~~Main Menu~~
==> 1. Insertion and Overflow Demo
==> 2. Deletion and Underflow Demo
==> 3. Display
==> 4. Exit
Enter Your Choice: **1**
Enter the element to be inserted: B

~~Main Menu~~
==> 1. Insertion and Overflow Demo
==> 2. Deletion and Underflow Demo
==> 3. Display
==> 4. Exit
Enter Your Choice: **1**
Enter the element to be inserted: C

~~Main Menu~~
==> 1. Insertion and Overflow Demo
==> 2. Deletion and Underflow Demo
==> 3. Display
==> 4. Exit
Enter Your Choice: **1**
Enter the element to be inserted: D
~~Circular Queue Overflow~~

~~Main Menu~~
==> 1. Insertion and Overflow Demo
==> 2. Deletion and Underflow Demo
==> 3. Display
==> 4. Exit
Enter Your Choice: **3**

Circular Queue contents are:
Front[0]-> A B C <-[2]Rear

~~Main Menu~~
==> 1. Insertion and Overflow Demo
==> 2. Deletion and Underflow Demo
==> 3. Display
==> 4. Exit
Enter Your Choice: **2**

Deleted element from the queue is: A

~~Main Menu~~
==> 1. Insertion and Overflow Demo
==> 2. Deletion and Underflow Demo
==> 3. Display
==> 4. Exit
Enter Your Choice: **3**
Circular Queue contents are:

Front[1]-> B C <-[2]Rear

~~Main Menu~~

==> 1. Insertion and Overflow Demo

==> 2. Deletion and Underflow Demo

==> 3. Display

==> 4. Exit

Enter Your Choice: **1**

Enter the element to be inserted: E

~~Main Menu~~

==> 1. Insertion and Overflow Demo

==> 2. Deletion and Underflow Demo

==> 3. Display

==> 4. Exit

Enter Your Choice: **3**

Circular Queue contents are:

Front[1]-> B C E <-[0]Rear

~~Main Menu~~

==> 1. Insertion and Overflow Demo

==> 2. Deletion and Underflow Demo

==> 3. Display

****

7.Design, Develop and Implement a menu driven Program in C for the following operations on Singly Linked List (SLL) of Student Data with the fields: USN, Name, Branch, Sem, PhNo

a. Create a SLL of N Students Data by using front insertion.

b. Display the status of SLL and count the number of nodes in it

c. Perform Insertion and Deletion at End of SLL

d. Perform Insertion and Deletion at Front of SLL

e. Demonstrate how this SLL can be used as STACK and QUEUE

f. Exit

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>

struct node
{
        char usn[25], name[25], branch[25];
        int sem;
        long int phoneNo;
        struct node *link;
};
typedef struct node * NODE;

NODE first = NULL;
int count = 0;

NODE createStudentNode()
{
        char us[20], nam[20], bran[20];
        int se;
        long int pn;
        printf("\nEnter the usn, Name, Branch, sem, PhoneNo of the student: \n");
        scanf("%s %s %s %d %ld", us, nam, bran, &se, &pn);
        NODE studentNode;
        studentNode = (NODE)malloc(sizeof(struct node));
        if(studentNode == NULL)
        {
                printf("\nMemory is not available");
                exit(0);
        }
        strcpy(studentNode->usn,   us);
        strcpy(studentNode->name,  nam);
        strcpy(studentNode->branch, bran);
        studentNode->sem = se;
        studentNode->phoneNo = pn;
        count++;
        return studentNode;
}
```

```
NODE insertAtFront()
{
        NODE temp;
        temp = createStudentNode();
        if(first == NULL)
        {
                temp->link = NULL;
                return temp;
         }
         temp->link = first;
         return temp;
}

NODE deleteAtFront()
{
        NODE temp;
        if(first == NULL)
        {
                printf("\nLinked list is empty");
                return NULL;
         }
         if(first->link == NULL)
         {
                 printf("\nThe Student node with usn:%s is deleted ", first->usn);
                 count--;
                 free(first);
                 return NULL;
          }
          temp = first;
          first = first->link;
          printf("\nThe Student node with usn:%s is deleted", temp->usn);
          count--;
          free(temp);
          return first;
}

NODE insertAtEnd()
{
          NODE cur, temp;
          temp = createStudentNode();
          temp->link = NULL;
          if(first == NULL)
          {
                  return temp;
          }
          if(first->link == NULL)
          {
                  first->link = temp;
                  return first;
          }
          cur = first;
```

```
        while(cur->link != NULL)
        {
                cur = cur->link;
        }
        cur->link = temp;
        return first;
}

NODE deleteAtEnd()
{
        NODE cur, prev;
        if(first == NULL)
        {
                printf("\nLinked List is empty");
                return NULL;
         }
         if(first->link == NULL)
         {
                 printf("\nThe student node with the usn:%s is deleted", first->usn);
                 free(first);
                 count--;
                  return NULL;
          }
          prev = NULL;
          cur = first;
          while(cur->link!=NULL)
          {
                prev = cur;
                cur = cur->link;
          }
          printf("\nThe student node with the usn:%s is deleted",cur->usn);
          free(cur);
          prev->link = NULL;
          count--;
          return first;
}

void displayStatus()
{
        NODE cur;
        int nodeNo = 1;
        cur = first;
        printf("\nThe contents of SLL: \n");
        if(cur == NULL)
                  printf("\nNo Contents to display in SLL \n");
        while(cur!=NULL)
        {
                  printf("\n||%d||", nodeNo);
                  printf(" USN:%s|", cur->usn);
                  printf(" Name:%s|",  cur->name);
                  printf(" Branch:%s|", cur->branch);
                  printf(" Sem:%d|", cur->sem);
```

```
                printf(" Ph:%ld|", cur->phoneNo);
                cur = cur->link;
                nodeNo++;
        }
        printf("\n No of student nodes is %d \n",count);
}

void stackDemoUsingSLL()
{
        int ch;
        while(1)
        {
                printf("\n~~~Stack Demo using SLL~~~\n");
                printf("\n1:Push operation \n2: Pop operation \n3: Display \n4:Exit \n");
                printf("\nEnter your choice for stack demo");
                scanf("%d", &ch);
                switch(ch)
                {
                        case 1:     first = insertAtFront();
                                    break;
                        case 2:     first = deleteAtFront();
                                    break;
                        case 3:     displayStatus();
                                    break;
                        default : return;
                }
         }
        return;
}

void main()
{
        int ch, i, n;
        while(1)
        {
                printf("\n~~~Menu~~~");
                printf("\nEnter your choice for SLL operation \n");
                printf("\n1:Create SLL of Student Nodes");
                printf("\n2:DisplayStatus");
                printf("\n3:InsertAtEnd");
                printf("\n4:DeleteAtEnd");
                printf("\n5:Stack Demo using SLL(Insertion and Deletion at Front)");
                printf("\n6:Exit \n");
                printf("\nEnter your choice:");
                scanf("%d", &ch);
                switch(ch)
                {
                        case 1 :    printf("\nEnter the no of students:    ");
                                    scanf("%d", &n);
                                    for(i=1; i<=n; i++)
                                            first = insertAtFront();
                                     break;
```

```
                    case 2:     displayStatus();
                                break;
                    case 3:     first = insertAtEnd();
                                break;
                    case 4:     first = deleteAtEnd();
                                break;

                    case 5:     stackDemoUsingSLL();
                                break;
                    case 6: exit(0);
                    default: printf("\nPlease enter the valid choice");
            }
        }
}
```

Output:

~~~Menu~~~
Enter your choice for SLL operation
1:Create SLL of Student Nodes
2:DisplayStatus
3:InsertAtEnd
4:DeleteAtEnd
5:Stack Demo using SLL(Insertion and Deletion at Front)
6:Exit
Enter your choice:**1**

Enter the no of students: 3

Enter the usn,Name,Branch, sem,PhoneNo of the student:
111
aaa
cs
1
111111

Enter the usn,Name,Branch, sem,PhoneNo of the student:
222
bbb
ec
2
222222

Enter the usn,Name,Branch, sem,PhoneNo of the student:
333
ccc
ec
3
333333

~~~Menu~~~

Enter your choice for SLL operation
1:Create SLL of Student Nodes
2:DisplayStatus
3:InsertAtEnd
4:DeleteAtEnd
5:Stack Demo using SLL(Insertion and Deletion at Front)
6:Exit
Enter your choice:**2**

The contents of SLL:

||1|| USN:333| Name:ccc| Branch:ec| Sem:3| Ph:333333|
||2|| USN:222| Name:bbb| Branch:ec| Sem:2| Ph:222222|
||3|| USN:111| Name:aaa| Branch:cs| Sem:1| Ph:111111|
No of student nodes is 3

~~~Menu~~~
Enter your choice for SLL operation

1:Create SLL of Student Nodes
2:DisplayStatus
3:InsertAtEnd
4:DeleteAtEnd
5:Stack Demo using SLL(Insertion and Deletion at Front)
6:Exit
Enter your choice:**3**

Enter the usn,Name,Branch, sem,PhoneNo of the student:
444
ddd
ec
4
444444

~~~Menu~~~
Enter your choice for SLL operation
1:Create SLL of Student Nodes
2:DisplayStatus
3:InsertAtEnd
4:DeleteAtEnd
5:Stack Demo using SLL(Insertion and Deletion at Front)
6:Exit
Enter your choice:**2**

The contents of SLL:

||1|| USN:333| Name:ccc| Branch:ec| Sem:3| Ph:333333|
||2|| USN:222| Name:bbb| Branch:ec| Sem:2| Ph:222222|
||3|| USN:111| Name:aaa| Branch:cs| Sem:1| Ph:111111|
||4|| USN:444| Name:ddd| Branch:ec| Sem:4| Ph:444444|
No of student nodes is 4

~~~Menu~~~
Enter your choice for SLL operation
1:Create SLL of Student Nodes
2:DisplayStatus
3:InsertAtEnd
4:DeleteAtEnd
5:Stack Demo using SLL(Insertion and Deletion at Front)
6:Exit
Enter your choice:**4**

The student node with the usn: 444 is deleted

~~~Menu~~~
Enter your choice for SLL operation
1:Create SLL of Student Nodes
2:DisplayStatus
3:InsertAtEnd
4:DeleteAtEnd
5:Stack Demo using SLL(Insertion and Deletion at Front)
6:Exit
Enter your choice:**2**

The contents of SLL:

||1|| USN:333| Name:ccc| Branch:ec| Sem:3| Ph:333333|
||2|| USN:222| Name:bbb| Branch:ec| Sem:2| Ph:222222|
||3|| USN:111| Name:aaa| Branch:cs| Sem:1| Ph:111111|
No of student nodes is 3

~~~Menu~~~
Enter your choice for SLL operation
1:Create SLL of Student Nodes
2:DisplayStatus
3:InsertAtEnd
4:DeleteAtEnd
5:Stack Demo using SLL(Insertion and Deletion at Front)
6:Exit
Enter your choice:**4**

The student node with the usn: 111 is deleted

~~~Menu~~~
Enter your choice for SLL operation
1:Create SLL of Student Nodes
2:DisplayStatus
3:InsertAtEnd
4:DeleteAtEnd
5:Stack Demo using SLL(Insertion and Deletion at Front)
6:Exit

Enter your choice:**5**

~~~Stack Demo using SLL~~~
1:Push operation
2: Pop operation
3: Display
4:Exit
Enter your choice for stack demo: **1**

Enter the usn,Name,Branch, sem,PhoneNo of the student:
555
eee
cs
1
555555

~~~Stack Demo using SLL~~~
1:Push operation
2: Pop operation
3: Display
4:Exit
Enter your choice for stack demo:**3**

The contents of SLL:

||1|| USN:555| Name:eee| Branch:cs| Sem:1| Ph:555555|
||2|| USN:333| Name:ccc| Branch:ec| Sem:3| Ph:333333|
||3|| USN:222| Name:bbb| Branch:ec| Sem:2| Ph:222222|
No of student nodes is 3

~~~Stack Demo using SLL~~~
1:Push operation
2: Pop operation
3: Display
4:Exit
Enter your choice for stack demo: **1**

Enter the usn,Name,Branch, sem,PhoneNo of the student:
666
fff
cs
6
666666

~~~Stack Demo using SLL~~~
1:Push operation
2: Pop operation
3: Display
4:Exit
Enter your choice for stack demo: **3**

The contents of SLL:

||1|| USN:666| Name:fff| Branch:cs| Sem:6| Ph:666666|
||2|| USN:555| Name:eee| Branch:cs| Sem:1| Ph:555555|
||3|| USN:333| Name:ccc| Branch:ec| Sem:3| Ph:333333|
||4|| USN:222| Name:bbb| Branch:ec| Sem:2| Ph:222222|
No of student nodes is 4

~~~Stack Demo using SLL~~~
1:Push operation
2: Pop operation
3: Display
4:Exit
Enter your choice for stack demo: **2**
The Student node with usn: 666 is deleted

~~~Stack Demo using SLL~~~
1:Push operation
2: Pop operation
3: Display
4:Exit
Enter your choice for stack demo: **3**

The contents of SLL:

||1|| USN:555| Name:eee| Branch:cs| Sem:1| Ph:555555|
||2|| USN:333| Name:ccc| Branch:ec| Sem:3| Ph:333333|
||3|| USN:222| Name:bbb| Branch:ec| Sem:2| Ph:222222|
No of student nodes is 3

~~~Stack Demo using SLL~~~
1:Push operation
2: Pop operation
3: Display
4:Exit
Enter your choice for stack demo: **4**

~~~Menu~~~
Enter your choice for SLL operation
1:Create SLL of Student Nodes
2:DisplayStatus
3:InsertAtEnd
4:DeleteAtEnd
5:Stack Demo using SLL(Insertion and Deletion at Front)
6:Exit
Enter your choice:**6**

****

8. Design, Develop and Implement a menu driven Program in C for the following operations on Doubly Linked List (DLL) of Employee Data with the fields: SSN, Name, Dept, Designation,Sal, PhNo

a. Create a DLL of N Employees Data by using end insertion.

b. Display the status of DLL and count the number of nodes in it

c. Perform Insertion and Deletion at End of DLL

d. Perform Insertion and Deletion at Front of DLL

e. Demonstrate how this DLL can be used as Double Ended Queue

f. Exit

****

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>

struct node
{
        int iSSN;
        char cName[30], cDept[4], cDesignation[30], cPhNo[11];
   int iSalary;
        struct node *plink;
        struct node *nlink;
};

typedef struct node* NODEPTR;

NODEPTR fnGetNode(void);
void fnFreeNode(NODEPTR);
NODEPTR fnInsRear(NODEPTR);
NODEPTR fnDelFront(NODEPTR);
NODEPTR fnInsFront(NODEPTR);
NODEPTR fnDelRear(NODEPTR);
void fnDisplay(NODEPTR);

int main()
```

```
{
        NODEPTR first = NULL;
        int iChoice, iNum, i;

   printf("nEnter the number of Employees N : "); scanf("%d", &iNum);
        for(i=0;i<iNum;i++)
        {
                printf("nEnter Data for Node %d :n", i+1);
                first = fnInsRear(first);
        }


        for(;;)
        {
                printf("nDLL OPERATIONSn");
                printf("====================");
                printf("n1.Insert Rearn2.Delete Frontn3.Insert Frontn4.Delete Rearn5.Displayn6.Exitn");
                printf("nEnter your choicen");
                scanf("%d",&iChoice);

                switch(iChoice)
                {
                        case 1: first = fnInsRear(first);
                                break;

                        case 2: first = fnDelFront(first);
                                break;

                        case 3: first = fnInsFront(first);
                                break;

                        case 4: first = fnDelRear(first);
                                break;

                        case 5: fnDisplay(first);
```

```
                                break;

                        case 6: exit(0);
                }
        }
   return 0;
}

NODEPTR fnGetNode()
{
        NODEPTR newborn;

        newborn = (NODEPTR)malloc(sizeof(struct node));

        if(newborn == NULL)
        {
                printf("nMemory Overflow");
                exit(0);
        }
        printf("nEnter SSN : ");
   scanf("%d",&newborn->iSSN);
   printf("nEnter name : ");
        scanf("%s",newborn->cName);
   printf("nEnter Department : ");
   scanf("%s", newborn->cDept);
   printf("nEnter Designation : ");
   scanf("%s", newborn->cDesignation);
        printf("nEnter Salary : ");
   scanf("%d",&newborn->iSalary);
        printf("nEnter Phone no : ");
   scanf("%s",newborn->cPhNo);

        return newborn;
}
```

```
void fnFreeNode(NODEPTR x)
{
        free(x);
}


NODEPTR fnInsRear(NODEPTR first)
{
        NODEPTR temp,cur;

        temp = fnGetNode();

        temp->plink = temp->nlink = NULL;

   if(first == NULL)
      return temp;
   cur = first;
   while(cur->nlink != NULL)
   {
      cur = cur->nlink;
   }
   cur->nlink = temp;
   temp->plink = cur;
   return first;
}

NODEPTR fnInsFront(NODEPTR first)
{
   NODEPTR temp;

        temp = fnGetNode();
   temp->plink = temp->nlink = NULL;

   temp->nlink = first;
```

```
   first = temp;
   return first;

}

NODEPTR fnDelRear(NODEPTR first)
{
        NODEPTR cur, prev;
        if(first == NULL)
        {
                printf("nDLL is emptyn");
                return first;
        }

        cur = first;
        if(cur->nlink == NULL)
        {
      printf("nNode deleted for %sn",cur->cName);
      fnFreeNode(cur);
      return NULL;
        }
   while(cur->nlink != NULL)
   {
      cur = cur->nlink;
   }

   prev = cur->plink;
   prev->nlink = NULL;
        printf("nNode deleted for %sn",cur->cName);
        fnFreeNode(cur);
        return first;
}

NODEPTR fnDelFront(NODEPTR first)
{
```

```
	NODEPTR temp;
	if(first == NULL)
	{
		printf("nDLL is emptyn");
		return first;
	}
	if(first->nlink == NULL)
	{
		printf("nNode deleted for %sn",first->cName);
		fnFreeNode(first);
		return NULL;
	}
	temp = first;
	first = first->nlink;
	first->plink = NULL;
	printf("nNode deleted for %sn",temp->cName);
	fnFreeNode(temp);
	return first;
}

void fnDisplay(NODEPTR first)
{
	NODEPTR curr;
	int count = 0;
	if(first == NULL)
	{
		printf("nDLL is emptyn");
		return;
	}

	printf("nThe contents of DLL are :n");
	curr = first;
//	printf("n");
   printf("nSSNtNametDepttDesignationtSalaryttPhone No");
	while(curr != NULL)
```

```
        {
                printf("n%-5dt%st%st%stt%-7dtt%-11s",curr->iSSN, curr->cName, curr->cDept, curr->cDesignation, curr->iSalary, curr->cPhNo);
                curr = curr->nlink;
                count++;
        }
        printf("nnDLL has %d nodesn", count);

}

/*CPP*/
```

Output

```
putta:~/.../Programs$ gcc -Wall 08_DLL.c
putta:~/.../Programs$ ./a.out

Enter the number of Employees N : 2

Enter Data for Node 1 :

Enter SSN : 1234

Enter name : Raju

Enter Department : CSE

Enter Designation : Clerk

Enter Salary : 34000

Enter Phone no : 8798987523

Enter Data for Node 2 :

Enter SSN : 1235
```

Enter name : Stan

Enter Department : PWD

Enter Designation : Driver

Enter Salary : 29000

Enter Phone no : 8232489410

DLL OPERATIONS

==========

1.Insert Rear

2.Delete Front

3.Insert Front

4.Delete Rear

5.Display

6.Exit

Enter your choice

5

The contents of DLL are :

SSN	Name	Dept	Designation	Salary	Phone No
1234	Raju	CSE	Clerk	34000	8798987523
1235	Stan	PWD	Driver	29000	8232489410

DLL has 2 nodes

DLL OPERATIONS

==========

1.Insert Rear

2.Delete Front

3.Insert Front

4.Delete Rear

5.Display

6.Exit

Enter your choice

1

Enter SSN : 1236

Enter name : Tara

Enter Department : EEE

Enter Designation : Peon

Enter Salary : 20000

Enter Phone no : 8397338933

DLL OPERATIONS

==========

1.Insert Rear

2.Delete Front

3.Insert Front

4.Delete Rear

5.Display

6.Exit

Enter your choice

3

Enter SSN : 1233

Enter name : Babu

Enter Department : RLW

Enter Designation : Manager

Enter Salary : 45000

Enter Phone no : 9956726282

DLL OPERATIONS

==========

1.Insert Rear

2.Delete Front

3.Insert Front

4.Delete Rear

5.Display

6.Exit

Enter your choice

5

The contents of DLL are :

SSN	Name	Dept	Designation	Salary	Phone No
1233	Babu	RLW	Manager	45000	9956726282
1234	Raju	CSE	Clerk	34000	8798987523
1235	Stan	PWD	Driver	29000	8232489410
1236	Tara	EEE	Peon	20000	8397338933

DLL has 4 nodes

DLL OPERATIONS

==========

1.Insert Rear

2.Delete Front

3.Insert Front

4.Delete Rear

5.Display

6.Exit

Enter your choice

4

Node deleted for Tara

DLL OPERATIONS

==========

1.Insert Rear

2.Delete Front

3.Insert Front

4.Delete Rear

5.Display

6.Exit

Enter your choice

5

The contents of DLL are :

SSN	Name	Dept	Designation	Salary	Phone No
1233	Babu	RLW	Manager	45000	9956726282
1234	Raju	CSE	Clerk	34000	8798987523
1235	Stan	PWD	Driver	29000	8232489410

DLL has 3 nodes

DLL OPERATIONS

==========

1.Insert Rear

2.Delete Front

3.Insert Front

4.Delete Rear

5.Display

6.Exit

Enter your choice

4

Node deleted for Stan

DLL OPERATIONS

==========

1.Insert Rear

2.Delete Front

3.Insert Front

4.Delete Rear

5.Display

6.Exit

Enter your choice

2

Node deleted for Babu

DLL OPERATIONS

==========

1.Insert Rear

2.Delete Front

3.Insert Front

4.Delete Rear

5.Display

6.Exit

Enter your choice

5

The contents of DLL are :

SSN	Name	Dept	Designation	Salary	Phone No
1234	Raju	CSE	Clerk	34000	8798987523

DLL has 1 nodes

DLL OPERATIONS

==========

1.Insert Rear

2.Delete Front

3.Insert Front

4.Delete Rear

5.Display

6.Exit

Enter your choice

2

Node deleted for Raju

DLL OPERATIONS

==========

1.Insert Rear

2.Delete Front

3.Insert Front

4.Delete Rear

5.Display

6.Exit

Enter your choice

2

DLL is empty

DLL OPERATIONS

==========

1.Insert Rear

2.Delete Front

3.Insert Front

4.Delete Rear

5.Display

6.Exit

Enter your choice

5

DLL is empty

DLL OPERATIONS

==========

1.Insert Rear

2.Delete Front

3.Insert Front

4.Delete Rear

5.Display

6.Exit

Enter your choice

6

****

9.Design, Develop and Implement a Program in C for the following operations on Singly Circular Linked List (SCLL) with header nodes

a. Represent and Evaluate a Polynomial P(x,y,z) = 6x2y2z-4yz5+3x3yz+2xy5z-2xyz3

b. Find the sum of two polynomials POLY1(x,y,z) and POLY2(x,y,z) and store the result in POLYSUM(x,y,z)

Support the program with appropriate functions for each of the above operations

****

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
#define COMPARE(x, y)      ( (x == y) ? 0 : (x > y) ? 1 : -1)

struct node
{
        int coef;
        int xexp, yexp, zexp;
        struct node *link;
};
typedef struct node *NODE;

NODE getnode()
{
        NODE x;
        x = (NODE) malloc(sizeof(struct node));
        if(x == NULL)
        {
                printf("Running out of memory \n");
                return NULL;
        }
        return x;
}
NODE attach(int coef, int xexp, int yexp, int zexp, NODE head)
{
        NODE temp, cur;
        temp = getnode();
        temp->coef = coef;
        temp->xexp = xexp;
        temp->yexp = yexp;
        temp->zexp = zexp;
        cur = head->link;
        while(cur->link != head)
        {
                cur = cur->link;
        }
        cur->link = temp;
        temp->link = head;
        return head;
}
```

```
NODE read_poly(NODE head)
{
        int i, j, coef, xexp, yexp, zexp, n;
        printf("\nEnter the no of terms in the polynomial: ");
        scanf("%d", &n);
        for(i=1; i<=n; i++)
        {
                printf("\n\tEnter the %d term: ",i);
                printf("\n\t\tCoef =  ");
                scanf("%d", &coef);
                printf("\n\t\tEnter Pow(x) Pow(y) and Pow(z): ");
                scanf("%d", &xexp);
                scanf("%d", &yexp);
                scanf("%d", &zexp);
                head = attach(coef, xexp, yexp, zexp, head);
        }
         return head;
}

void display(NODE head)
{
        NODE temp;
        if(head->link == head)
        {
                printf("\nPolynomial does not exist.");
                return;
        }
        temp = head->link;

        while(temp != head)
        {
                printf("%dx^%dy^%dz^%d", temp->coef, temp->xexp, temp->yexp, temp->zexp);
                temp = temp->link;
                if(temp != head)
                        printf(" + ");
        }
}

int poly_evaluate(NODE head)
{
        int x, y, z, sum = 0;
        NODE poly;

        printf("\nEnter the value of x,y and z: ");
        scanf("%d %d %d", &x, &y, &z);

        poly = head->link;
        while(poly != head)
        {
                sum += poly->coef * pow(x,poly->xexp)* pow(y,poly->yexp) * pow(z,poly->zexp);
                poly = poly->link;
        }
```

```
        return sum;
}

NODE poly_sum(NODE head1, NODE head2, NODE head3)
{
   NODE a, b;
   int coef;
   a = head1->link;
   b = head2->link;

   while(a!=head1 && b!=head2)
   {
        while(1)
        {
                if(a->xexp == b->xexp && a->yexp == b->yexp && a->zexp == b->zexp)
                {
                        coef = a->coef + b->coef;
                        head3 = attach(coef, a->xexp, a->yexp, a->zexp, head3);
                        a = a->link;
                        b = b->link;
                        break;
                } //if ends here
                if(a->xexp!=0 || b->xexp!=0)
                {
                        switch(COMPARE(a->xexp, b->xexp))
                        {
                         case -1 :  head3 = attach(b->coef, b->xexp, b->yexp, b->zexp, head3);
                                    b = b->link;
                                    break;

                        case 0 :    if(a->yexp > b->yexp)
                                    {
                                        head3 = attach(a->coef, a->xexp, a->yexp, a->zexp, head3);
                                        a = a->link;
                                        break;
                                     }
                                     else if(a->yexp < b->yexp)
                                     {
                                         head3 = attach(b->coef, b->xexp, b->yexp, b->zexp, head3);
                                         b = b->link;
                                         break;
                                     }
                                    else if(a->zexp > b->zexp)
                                    {
                                        head3 = attach(a->coef, a->xexp, a->yexp, a->zexp, head3);
                                        a = a->link;
                                        break;
                                    }
                                    else if(a->zexp < b->zexp)
                                   {
                                        head3 = attach(b->coef, b->xexp, b->yexp, b->zexp, head3);
                                        b = b->link;
```

```
                                    break;
                                }
                        case 1 :    head3 = attach(a->coef,a->xexp,a->yexp,a->zexp,head3);
                                    a = a->link;
                                    break;
                        } //switch ends here
                        break;
                } //if ends here
                if(a->yexp!=0 || b->yexp!=0)
                {
                        switch(COMPARE(a->yexp, b->yexp))
                        {
                             case -1 :  head3 = attach(b->coef, b->xexp, b->yexp, b->zexp, head3);
                                        b = b->link;
                                        break;
                              case 0 :  if(a->zexp > b->zexp)
                                        {
                                             head3 = attach(a->coef, a->xexp, a->yexp, a->zexp, head3);
                                             a = a->link;
                                             break;
                                         }
                                         else if(a->zexp < b->zexp)
                                         {
                                             head3 = attach(b->coef, b->xexp, b->yexp, b->zexp, head3);
                                             b = b->link;
                                             break;
                                         }
                                 case 1 :   head3 = attach(a->coef, a->xexp, a->yexp, a->zexp, head3);
                                            a = a->link;
                                            break;
                        }
                        break;
                }
                if(a->zexp!=0 || b->zexp!=0)
                {
                        switch(COMPARE(a->zexp,b->zexp))
                        {
                                case -1 :   head3 = attach(b->coef,b->xexp,b->yexp,b->zexp,head3);
                                            b = b->link;
                                            break;
                                case 1 :    head3 = attach(a->coef, a->xexp, a->yexp, a->zexp, head3);
                                            a = a->link;
                                            break;
                        }
                        break;
                }
        }
}
while(a!= head1)
{
        head3 = attach(a->coef,a->xexp,a->yexp,a->zexp,head3);
       a = a->link;
```

```
    }
    while(b!= head2)
    {
          head3 = attach(b->coef,b->xexp,b->yexp,b->zexp,head3);
          b = b->link;
    }
    return head3;
}
void main()
{
  NODE head, head1, head2, head3;
  int res, ch;
  head =  getnode();    /* For polynomial evalaution */
  head1 = getnode();    /* To hold POLY1 */
  head2 = getnode();    /* To hold POLY2 */
  head3 = getnode();    /* To hold POLYSUM */

  head->link=head;
  head1->link=head1;
  head2->link=head2;
  head3->link= head3;

  while(1)
  {
      printf("\n~~~Menu~~~");
      printf("\n1.Represent and Evaluate a Polynomial P(x,y,z)");
      printf("\n2.Find the sum of two polynomials POLY1(x,y,z)");
      printf("\nEnter your choice:");
      scanf("%d",&ch);
      switch(ch)
      {
         case 1:        printf("\n~~~~Polynomial evaluation P(x,y,z)~~~\n");
                        head = read_poly(head);
                        printf("\nRepresentation of Polynomial for evaluation: \n");
                        display(head);
                        res = poly_evaluate(head);
                        printf("\nResult of polynomial evaluation is : %d \n", res);
                        break;

         case 2:        printf("\nEnter the POLY1(x,y,z):  \n");
                        head1 = read_poly(head1);
                        printf("\nPolynomial 1 is:  \n");
                        display(head1);

                        printf("\nEnter the POLY2(x,y,z):  \n");
                        head2 = read_poly(head2);
                        printf("\nPolynomial 2 is: \n");
                        display(head2);

                        printf("\nPolynomial addition result: \n");
                        head3 = poly_sum(head1,head2,head3);
                        display(head3);
```

```
                    break;
            case 3: exit(0);
        }
    }
}
```

Output:

~~~Menu~~~
1.Represent and Evaluate a Polynomial P(x,y,z)
2.Find the sum of two polynomials POLY1(x,y,z) and POLY2(x,y,z)
Enter your choice: **1**

~~~~Polynomial evaluation P(x,y,z)~~~
Enter the no of terms in the polynomial: **5**
Enter the 1 term:
Coef = **6**
Enter Pow(x) Pow(y) and Pow(z): **2 2 1**
Enter the 2 term:
Coef = **-4**
Enter Pow(x) Pow(y) and Pow(z): **0 1 5**
Enter the 3 term:
Coef = **3**
Enter Pow(x) Pow(y) and Pow(z): **3 1 1**
Enter the 4 term:
Coef = **2**
Enter Pow(x) Pow(y) and Pow(z): **1 5 1**
Enter the 5 term:
Coef = **-2**
Enter Pow(x) Pow(y) and Pow(z): **1 1 3**
Representation of Polynomial for evaluation:
6x^2y^2z^1 + -4x^0y^1z^5 + 3x^3y^1z^1 + 2x^1y^5z^1 + -2x^1y^1z^3
Enter the value of x,y and z: **1 1 1**
Result of polynomial evaluation is : **5**

~~~Menu~~~
1.Represent and Evaluate a Polynomial P(x,y,z)
2.Find the sum of two polynomials POLY1(x,y,z) and POLY2(x,y,z)
Enter your choice: **2**

Enter the POLY1(x,y,z):
Enter the no of terms in the polynomial: **5**
Enter the 1 term:
Coef = **6**
Enter Pow(x) Pow(y) and Pow(z): **4 4 4**
Enter the 2 term:
Coef = **3**
Enter Pow(x) Pow(y) and Pow(z): **4 3 1**
Enter the 3 term:
Coef = **5**
Enter Pow(x) Pow(y) and Pow(z): **0 1 1**
Enter the 4 term:
Coef = **10**

Enter Pow(x) Pow(y) and Pow(z): **0 1 0**

Enter the 5 term:

Coef = **5**

Enter Pow(x) Pow(y) and Pow(z): **0 0 0**

Polynomial 1 is:

6x^4y^4z^4 + 3x^4y^3z^1 + 5x^0y^1z^1 + 10x^0y^1z^0 + 5x^0y^0z^0

Enter the POLY2(x,y,z):

Enter the no of terms in the polynomial: **5**

Enter the 1 term:

Coef = **8**

Enter Pow(x) Pow(y) and Pow(z): **4 4 4**

Enter the 2 term:

Coef = **4**

Enter Pow(x) Pow(y) and Pow(z): **4 2 1**

Enter the 3 term:

Coef = **30**

Enter Pow(x) Pow(y) and Pow(z): **0 1 0**

Enter the 4 term:

Coef = **20**

Enter Pow(x) Pow(y) and Pow(z): **0 0 1**

Enter the 5 term:

Coef = **3**

Enter Pow(x) Pow(y) and Pow(z): **0 0 0**

Polynomial 2 is:

8x^4y^4z^4 + 4x^4y^2z^1 + 30x^0y^1z^0 + 20x^0y^0z^1 + 3x^0y^0z^0

Polynomial addition result:

14x^4y^4z^4 + 3x^4y^3z^1 + 4x^4y^2z^1 + 5x^0y^1z^1 + 40x^0y^1z^0 + 20x^0y^0z^1 + 8x^0y^0z^0

~~~Menu~~~

1.Represent and Evaluate a Polynomial P(x,y,z)

2.Find the sum of two polynomials POLY1(x,y,z) and POLY2(x,y,z)

Enter your choice:3
~~~

****

10.Design, Develop and Implement a menu driven Program in C for the following operations on Binary Search Tree (BST) of Integers

a. Create a BST of N Integers: 6, 9, 5, 2, 8, 15, 24, 14, 7, 8, 5, 2

b. Traverse the BST in Inorder, Preorder and Post Order

c. Search the BST for a given element (KEY) and report the appropriate message

d. Delete an element(ELEM) from BST

e. Exit

****

```
#include<stdio.h>
#include<stdlib.h>
struct BST
{
        int data;
        struct BST *lchild;
        struct BST *rchild;
};
typedef struct BST * NODE;

NODE get_node()
{
        NODE temp;
        temp = (NODE) malloc(sizeof(struct BST));
        temp->lchild = NULL;
        temp->rchild = NULL;
        return temp;
}

void insert(NODE root, NODE newnode);
void inorder(NODE root);
void preorder(NODE root);
void postorder(NODE root);
void search(NODE root, int key);

void insert(NODE root, NODE newnode)
{
   /*Note: if newnode->data ==  root->data it will be skipped. No duplicate nodes are allowed */

        if (newnode->data < root->data)
        {
                if (root->lchild == NULL)
                        root->lchild = newnode;
                else
                        insert(root->lchild, newnode);
        }
        if (newnode->data > root->data)
        {
                if (root->rchild == NULL)
```

```
                root->rchild = newnode;
            else
                insert(root->rchild, newnode);
        }
}


void search(NODE root, int key)
{
        NODE cur;
        if(root == NULL)
        {
            printf("\nBST is empty.");
            return;
        }
        cur = root;
        while (cur != NULL)
        {
            if (cur->data == key)
            {
                printf("\nKey element %d is present in BST", cur->data);
                return;
            }
            if (key < cur->data)
                cur = cur->lchild;
            else
                cur = cur->rchild;
        }
        printf("\nKey element %d is not found in the BST", key);
}

void inorder(NODE root)
{
        if(root != NULL)
        {
            inorder(root->lchild);
            printf("%d ", root->data);
            inorder(root->rchild);
        }
}

void preorder(NODE root)
{
        if (root != NULL)
        {
            printf("%d ", root->data);
            preorder(root->lchild);
            preorder(root->rchild);
        }
}

void postorder(NODE root)
```

```
{
        if (root != NULL)
        {
                postorder(root->lchild);
                postorder(root->rchild);
                printf("%d ", root->data);
        }
}


void main()
{
        int ch, key, val, i, n;
        NODE root = NULL, newnode;
        while(1)
        {
                printf("\n~~~~BST MENU~~~~");
                printf("\n1.Create a BST");
                printf("\n2.Search");
                printf("\n3.BST Traversals: ");
                printf("\n4.Exit");
                printf("\nEnter your choice: ");
                scanf("%d", &ch);
                switch(ch)
                {
                        case 1:         printf("\nEnter the number of elements: ");
                                        scanf("%d", &n);
                                        for(i=1;i<=n;i++)
                                        {
                                                newnode = get_node();
                                                printf("\nEnter The value: ");
                                                scanf("%d", &val);
                                                newnode->data = val;
                                                if (root == NULL)
                                                        root = newnode;
                                                else
                                                        insert(root, newnode);
                                        }
                                        break;
                         case 2:        if (root == NULL)
                                                printf("\nTree Is Not Created");
                                        else
                                        {
                                                printf("\nThe Preorder display : ");
                                                preorder(root);
                                                printf("\nThe Inorder display : ");
                                                inorder(root);
                                                printf("\nThe Postorder display : ");
                                                postorder(root);
                                        }

                                        break;
```

```
                    case 3:     printf("\nEnter Element to be searched: ");
                                scanf("%d", &key);
                                search(root, key);
                                break;

                    case 4:  exit(0);
                }
        }
}
```

Output:

```
~~~~BST MENU~~~~
1.Create a BST
2.Search
3.BST Traversals:
4.Exit
Enter your choice: 1

Enter the number of elements: 12
Enter The value: 6
Enter The value: 9
Enter The value: 5
Enter The value: 2
Enter The value: 8
Enter The value: 15
Enter The value: 24
Enter The value: 14
Enter The value: 7
Enter The value: 8
Enter The value: 5
Enter The value: 2

~~~~BST MENU~~~~
1.Create a BST
2.Search
3.BST Traversals:
4.Exit
Enter your choice: 3

The Preorder display:   6  5  2  9  8  7  15  14  24
The Inorder display:    2  5  6  7  8  9  14  15  24
The Postorder display:  2  5  7  8  14  24  15  9  6

~~~~BST MENU~~~~
1.Create a BST
2.Search
3.BST Traversals:
4.Exit
Enter your choice: 2

Enter Element to be searched: 66
Key element 66 is not found in the BST
```

~~~~BST MENU~~~~
1.Create a BST
2.Search
3.BST Traversals:
4.Exit
Enter your choice: 2

Enter Element to be searched: 14
Key element 14 is present in BST

~~~~BST MENU~~~~
1.Create a BST
2.Search
3.BST Traversals:
4.Exit
Enter your choice: 4

****

11.Design, Develop and Implement a Program in C for the following operations on Graph(G) of Cities

a. Create a Graph of N cities using Adjacency Matrix.

b. Print all the nodes reachable from a given starting node in a digraph using BFS method

c. Check whether a given graph is connected or not using DFS method.

****

Program

```
#include<stdio.h>
#include<stdlib.h>
void BFS(int [20][20],int,int [20],int);
void DFS(int [20][20],int,int [20],int);
void main()
{
    int n,a[20][20],i,j,visited[20],source;
    printf("Enter the number of vertices:");
    scanf("%d",&n);
    printf("\nEnter the adjacency matrix:\n");
    for(i=1;i<=n;i++)
        for(j=1;j<=n;j++)
            scanf("%d",&a[i][j]);
    for(i=1;i<=n;i++)
        visited[i]=0;
    printf("\nEnter the source node:");
    scanf("%d",&source);
    visited[source]=1;
    BFS(a,source,visited,n);
    for(i=1;i<=n;i++)
    {
        if(visited[i]!=0)
            printf("\n Node %d is reachable",i);
        else
            printf("\n Node %d is not reachable",i);
    }
   DFS(a,source,visited,n);
    for(i=1;i<=n;i++)
    {
        if(visited[i]==0)
       {
            printf("\nGraph is not connected");

            exit(0);
        }
    }
    printf("\nGraph is connected\n");
}
void BFS(int a[20][20],int source,int visited[20],int n)
{
    int queue[20],f,r,u,v;
    f=0;
```

```
    r=-1;
    queue[++r]=source;
    while(f<=r)
    {
        u=queue[f++];
        for(v=1;v<=n;v++)
        {
            if(a[u][v]==1 && visited[v]==0)
            {
            queue[++r]=v;
            visited[v]=1;
          }
      }
  }
}
    void DFS(int a[20][20],int u,int visited[20],int n)
{
    int v;
    visited[u]=1;
    for(v=1;v<=n;v++)
    {
        if(a[u][v]==1 && visited[v]==0)
            DFS(a,v,visited,n);
    }
}
```

Sample Output:

```
[exam@cpl1-8 ~]$ cc ds11.c
[exam@cpl1-8 ~]$ ./a.out
Enter the number of vertices:4

Enter the adjacency matrix:
0 1 0 1
1 0 1 0
0 1 0 1
1 0 1 0

Enter the source node:1
 Node 1 is reachable
 Node 2 is reachable
 Node 3 is reachable
 Node 4 is reachable
Graph is connected
```

****

12.Given a File of N employee records with a set K of Keys(4-digit) which uniquely determine the records in file F. Assume that file F is maintained in memory by a Hash Table(HT) of m memory locations with L as the set of memory addresses (2-digit) of locations in HT. Let the keys in K and addresses in L are Integers. Design and develop a Program in C that uses Hash function H: K →L as H(K)=K mod m (remainder method), and implement hashing technique to map a given key K to the address space L. Resolve the collision (if any) using linear probing

****

```
#include<stdio.h>
#include<stdlib.h>

int key[20], n, m;
int *ht, hashindex;
int elecount = 0;

void createHashTable()
{
        int i;
        ht = (int*)malloc(m*sizeof(int));

        if(ht == NULL)
                printf("\nUnable to create the hash table");
        else
                for(i=0; i<m; i++)
                        ht[i] = -1;
 }

void insertIntoHashTable(int key)
{
        hashindex = key % m;

        while(ht[hashindex] != -1)
        {
                hashindex = (hashindex+1)%m;
        }
        ht[hashindex] = key;
        elecount++;
 }

void displayHashTable()
{
        int i;
        if(elecount == 0)
        {
                printf("\nHash Table is empty");
                return;
        }
```

```
        printf("\nHash Table contents are:\n\n ");
        for(i=0; i<m; i++)
                printf("\nT[%d] --> %d ", i, ht[i]);
}

void main()
{
        int i;
        printf("\nEnter the number of employee  records (N) :   ");
        scanf("%d", &n);

        printf("\nEnter the four digit key values (K) of 'N' Employee Records:\n  ");
        for(i=0; i<n; i++)
        scanf("%d", &key[i]);

        printf("\nEnter the two digit memory locations (m) for hash table:   ");
        scanf("%d", &m);

        createHashTable();

        printf("\nInserting key values of Employee records into hash table….. ");
        for(i=0; i<n; i++)
        {
                if(elecount == m)
                {
                        printf("\nHash table is full. Cannot insert the %d record key value", i+1);
                        break;
                }
                insertIntoHashTable(key[i]);
        }

        //Displaying Keys inserted into hash table
        displayHashTable();
}
```

Output:

Enter the number of employee records (N) : **12**

Enter the four digit key values (K) of 'N' Employee Records:

1234
5678
3456
2345
6799
1235
7890
3214
3456
1235
5679
2346

Enter the two digit memory locations (m) for hash table: **15**

Inserting key values of Employee records into hash table
Hash Table contents are:

T[0] --> 7890
T[1] --> -1
T[2] --> -1
T[3] --> -1
T[4] --> 1234
T[5] --> 2345
T[6] --> 3456
T[7] --> 6799
T[8] --> 5678
T[9] --> 1235
T[10] --> 3214
T[11] --> 3456
T[12] --> 1235
T[13] --> 5679
T[14] --> 2346