**RNS INSTITUTE OF TECHNOLOGY**

**Dr. VISHNUVARDHAN ROAD, CHANNASANDRA,BENGALURU-560 098**

**Department of Information Science & Engineering**



**DATA STRUCTURES MANUAL**

**III Semester**

**BCSL305**

**Faculty-in-charge**

**Dr. Sunitha K**

**Vinutha G K**

**Janardhan Singh**

**RNS INSTITUTE OF TECHNOLOGY**

**Dr. VISHNUVARDHAN ROAD, CHANNASANDRA, BENGALURU -560 098**

**Department of Information Science and Engineering**



**VISION of the College**

Building RNSIT into a World - Class Institution

**MISSION of**

**To impart high quality education** in Engineering, Technology and Management with a difference, enabling students to excel in their career by

1. Attracting quality Students and preparing them with a strong foundation in fundamentals so as *to achieve distinctions in various walks of life* leading to outstanding contributions.
2. Imparting value based, need based, and choice based and skill based professional education to the aspiring youth and *carving them into disciplined, World class Professionals* with *social responsibility.*
3. Promoting excellence in Teaching, Research and Consultancy that galvanizes academic consciousness among Faculty and Students.
4. Exposing Students to emerging frontiers of knowledge in various domains and make them suitable for Industry, Entrepreneurship, Higher studies, and Research & Development.
5. Providing freedom of action and choice for all the Stake holders with better visibility.

**VISION of the Department**

Building Information Technology Professionals by Imparting Quality Education and Inculcating Key Competencies.

**MISSION of the Department**

* Provide strong fundamentals through learner centric approach
* Instil technical, interpersonal, interdisciplinary skills and logical thinking for holistic development
* Train to excel in higher education, research, and innovation with global perspective
* Develop leadership and entrepreneurship qualities with societal responsibilities

**PROGRAM EDUCATIONAL OBJECTIVES (PEOs)**

**ISE Graduates** within three-four years of graduation will have

* **PEO1**: Acquired the fundamentals of computers and applied knowledge of Information Science & Engineering and continue to develop their technical competencies by problem solving using programming.
* **PEO2**: Ability to formulate problems, attained the Proficiency to develop system/application software in a scalable and robust manner with various platforms, tools and frameworks to provide cost effective solutions.
* **PEO3**: Obtained the capacity to investigate the necessities of the software Product, adapt to technological advancement, promote collaboration and interdisciplinary activities, Protecting Environment and developing Comprehensive leadership.
* **PEO4**: Enabled to be employed and provide innovative solutions to real-world problems across different domains.
* **PEO5**: Possessed communication skills, ability to work in teams, professional ethics, social responsibility, entrepreneur and management, to achieve higher career goals, and pursue higher studies.

**PROGRAM OUTCOMES(POs)**

**Engineering Graduates** will be ableto:

* **PO1:  Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization for the solution of complex engineering problems
* **PO2:  Problem analysis:** Identify, formulate, 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 public health and safety, and 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 the 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 and 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 life-long learning in the broadest context of technological change.

#### PROGRAM SPECIFIC OUTCOMES(PSOs)

#### ISE Graduateswill have

* **PSO1 – Problem Solving Abilities:** Ability to demonstrate the fundamental and theoretical concepts, analyze the real time problems and develop customized software solutions by applying the knowledge of mathematics and algorithmic techniques.
* **PSO2 – Applied Engineering Skills:** Enable creative thinking, Ability to apply standard practices and strategies, technical skills in software design, development, integration of systems and management for improving the security, reliability and survivability of the infrastructure.
* **PSO3 – General Expertise and Higher Learning:** Ability to exchange knowledge effectively demonstrate the ability of team work, documentation skills, professional ethics, entrepreneurial skills and continuing higher education in the field of Information technology.

**RNS INSTITUTE OF TECHNOLOGY**

**Dr. VISHNUVARDHAN ROAD, CHANNASANDRA, BENGALURU -560 098**

**Department of Information Science and Engineering**

**Data Structures and Applications**

**Subject Code: BCSL305 Total Hours: 28**

|  |  |
| --- | --- |
| **Scheme** | **NEP**  **(22 SCHEME)** |
| **I.A. Marks** | **50** |

**Hours/Week: 2P Exam Hours: 03**

**Course objectives**

This laboratory course enables students to get practical experience in design, develop, implement, analyze and evaluation/testing of

● Dynamic memory management

● Linear data structures and their applications such as stacks, queues and lists

● Non-Linear data structures and their applications such as trees and graphs.

**Course Outcomes**

After studying this course, students will be able to:

|  |  |
| --- | --- |
| CO1 | Analyse 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 given scenario. |
| CO4 | Apply the appropriate data structure for solving real world problems. |

**CO mapping to PO/PSOs**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **CO /** | **PO1** | **PO2** | **PO3** | **PO4** | **PO5** | **PO6** | **PO7** | **PO8** | **PO9** | **PO10** | **PO11** | **PO12** | **PSO1** | **PSO2** | **PSO3** |
| **PO & PSO** |
| **CO** 1 | 2 | 3 | 3 | 2 | 3 |  |  |  |  |  |  | 1 | 3 | 2 |  |
| **CO**2 | 3 | 3 | 3 | 2 | 3 |  |  |  |  |  |  | 2 | 3 | 3 |  |
| CO3 | 3 | 3 | 3 | 2 | 3 |  |  |  |  |  |  | 2 | 3 | 3 |  |
| CO4 | 3 | 3 | 3 | 3 | 3 |  |  |  |  |  |  | 2 | 2 | 3 |  |

**Data Structures Laboratory**

**Evaluation Rubrics**

**Subject Code: BSCL205 I.A. Marks : 40**

**Hours/Week: 01I+2P Exam Hours: 03**

**Total Hours: 28 Exam Marks: 50**

**Lab Write-up and Execution Rubrics (Max: 24 marks)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | **Above Average** | **Average** | **Below Average** |
| a. | **Understanding of problem and approach to solve.**  **(8 Marks)** | Able to analyze the given problem and efficiently implement using suitable high-level language instructions.(8-6) | Able to analyze the problem and moderate understanding of high-level language instructions.  (5-3) | Poor understanding of high-level language instructions or No program write-up.  (2-0) |
| b. | **Execution and Viva (5 questions)**  **(8 Marks)** | Program executed for varied inputs with valid results and able to answer allfive questions appropriately.(8-6) | Program is executed for some inputs and able to answer three-two questions. (5-3) | Program has compilation errors or no Execution and not answered any questions.  (2-0) |
| c. | **Results and Documentation**  **(8 Marks)** | Program and results obtained arelegibly written / documented.  (8-6) | Program and results obtained is acceptably documented.  (5-3) | No Proper results and poor documentation.  (2-0) |

**LABInternal Assessment rubrics (Max: 16 marks)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | **Above Average** | **Average** | **Below Average** |
| a. | **Write-up**  **(3 Marks)** | Able to write the complete code.(3) | Able to write the code with few errors. (2-1) | Unable to write.  (0) |
| b. | **Execution**  **(10 Marks)** | Executed successfully for all the inputs given. (10-7) | Obtained partially correct results.  (6-3) | Program has compilation errors or No Execution.  (2-0) |
| c. | **Viva (5 questions)**  **(3 Marks)** | Able to answer all five questions correctly. (3) | Able to answer three-two questions. (2-1) | Not answered any.(0) |

**Mini-project Assessment rubrics (Max: 10 marks)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | | **Above Average** | **Average** | **Below Average** | |
| a. | **Mini-project**  **(10 Marks)** | | Able to develop a mini-project completely and given presentation along with the team members. (10-7) | Able to develop a mini-project partially and given presentation along with the team members. (6-3) | Not able to develop mini-project or absent for the presentation. (2-0) | |
| **Sl. No** | | | **List of Programs** | | | | **Page**  **no** | |
| 1 | | | Develop a Program in C for the following:  a) Declare a calendar as an array of 7 elements (A dynamically Created array) to represent7 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. | | | | 1 | |
| 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 inSTR with REP if PAT exists in STR. Report suitable messages in case PAT does notexist in STR Support the program with functions for each of the above operations. Don't use Built-in functions. | | | | 4 | |
| 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. | | | | 6 | |
| 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. | | | | 9 | |
| 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 | | | | 11 | |
| 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. | | | | 15 | |
| 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 | | | | 23 | |
| **Sl. No** | | | **List of Programs** | | | | **Page**  **no** | |
| 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 | | | | 30 | |
| 9 | | | Develop 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. | | | | 30 | |
| 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 | | | | 36 | |
| 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. | | | | 42 | |
| 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. | | | | 45 | |

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>

typedef struct

{

char \*day;

int date;

char \*activity;

} calendar;

calendar\* create()

{

calendar \*week;

week=(calendar\*)calloc(7,sizeof(calendar));

return week;

}

void read(calendar \*week)

{

int i;

char day[20],activity[50];

printf("Enter week details(week day,date, activity)");

for(i=0;i<7;i++)

{

printf("Day %d:",i+1);

scanf("%s%d%s",day,&week[i].date,activity);

week[i].day = strdup(day);

week[i].activity = strdup(activity);

}

}

void display(calendar \*week)

{

int i;

printf("Week activity\nDay\tDate\tActivity\n");

for(i=0;i<7;i++)

{

printf("%s\t%d\t%s\n",week[i].day,week[i].date,week[i].activity);

}

}

int main()

{

int choice;

calendar \*week;

while(1)

{

printf("1.Create 2.Read 3.Display 4.Exit choice:");

scanf("%d",&choice);

switch(choice)

{

case 1: week = create();

if(week!=NULL)

printf("Calander is successfully created\n");

break;

case 2: read(week);

break;

case 3: display(week);

break;

case 4: return 0;

default: printf("Invalid choice\n");

}

}

}

**Sample input and output:**

1.Create 2.Read 3.Display 4.Exit choice:1

Calander is successfully created

1.Create 2.Read 3.Display 4.Exit choice:2

Enter week details(week day,date, activity)Day 1:Monday

11

College

Day 2:Tuesday

12

Assignment

Day 3:Wednesday

13

Projects

Day 4:Thursday

14

Coding

Day 5:Friday

15

Dance

Day 6:Saturday

16

Music

Day 7:Sunday

17

Relax

1.Create 2.Read 3.Display 4.Exit choice:3

Week activity

Day Date Activity

Monday 11 College

Tuesday 12 Assignment

Wednesday 13 Projects

Thursday 14 Coding

Friday 15 Dance

Saturday 16 Music

Sunday 17 Relax

1.Create 2.Read 3.Display 4.Exit choice:4

...Program finished with exit code 0

Press ENTER to exit console.

1. **Design, Develop and Implement a Program in C for the following operations on Strings**
2. **Read a main String (STR), a Pattern String (PAT) and a Replace String (REP) .**
3. **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.**

#include<stdio.h>

char str[100],pat[100],rep[100],ans[100];

void read()

{

printf("enter the string: ");

gets(str);

printf(" \n enter the patter string: ");

flushall();

gets(pat);

printf("\n enter the replacement string: ");

flushall();

gets(rep);

}

void pat\_match()

{

int i,j,c,m,k;

int flag=0;

i=m=c=j=0;

while(str[c]!='\0')

{

if(str[m]==pat[i])//Pattern matching

{

i++;

m++;

if(pat[i]=='\0')

{

printf("\n pat:%s is found at position %d",pat,c);

for(k=0;rep[k]!='\0';k++,j++)

ans[j]=rep[k];

i=0;

c=m;

flag=1;

}

}

else//pattern mismatch

{

ans[j]=str[c];

j++;

c++;

m=c;

i=0;

}

}

ans[j]='\0';

if(flag==0)

printf("\n PAT:%s is not found in STR:%s",pat,str);

else

printf("\n The resulting string is: %s",ans);

}

void main()

{

clrscr();

read();

pat\_match();

getch();

}

**OUTPUT:-**

enter the string: hi rns hi

enter the pattern string: hi

enter the replacement string: hello

pat:hi is found at position 0

pat:hi is found at position 7

The resulting string is: hello rns hello

enter the string: hi rns hi

enter the pattern string: rnsit

enter the replacement string: sjbit

PAT: rnsit is not found in STR: hi rns hi

**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**

#include<stdio.h>

#include<math.h>

#define max 5

int stack\_list[max],top=-1;

void push(int m)

{

if(top==max-1)

printf("\n Stack overflow");

else

{

top++;

stack\_list[top]=m;

}

}

int pop()

{

if(top==-1)

{

printf("\n Stack underflow");

return -1;

}

else

return stack\_list[top--];

}

void display()

{

int i;

if(top==-1)

printf("\n Stack is empty");

else

{

printf("\n The elements are\n");

for(i=top;i>=0;i--)

printf("%d\n",stack\_list[i]);

}

}

void palindrome()

{

int n,num,rem,i;

printf("\n Enter n");

scanf("%d",&n);

top=-1;

num=n;

while(num!=0)

{

rem=num%10;

push(rem);

num=num/10;

}

num=0;

for(i=0;top!=-1;i++)

num=pop()\*pow(10,i)+num;

if(n==num)

printf("\n It is a palindrome");

else

printf("\n It is not a palindrome");

}

int main()

{

int c,m;

while(1)

{ printf("\n Enter 1-push\n2-pop\n3-display\n4-palindrome");

scanf("%d",&c);

switch(c)

{

case 1: printf("\n Enter an element\t");

scanf("%d",&m);

push(m);

break;

case 2:m=pop();

if(m!=-1)

printf("\n The popped elementis %d",m);

break;

case 3: display();

break;

case 4: palindrome();

break;

default:return 0;

}

}

}

Sample Input & Output:

Enter your choice

1-push

2-pop

3-display

4-palindrome

1

Enter an element to be pushed 10

Enter your choice

1-push

2-pop

3-display

4-palindrome1

Enter an element to be pushed 20

Enter your choice

1-push

2-pop

3-display

4-palindrome1

Enter an element to be pushed 30

Enter your choice

1-push

2-pop

3-display

4-palindrome

1

Enter an element to be pushed 40

Stack overflow

Enter your choice

1-push

2-pop

3-display

4-palindrome

3

The elements are

30

20

10

Enter your choice

1-push

2-pop

3-display

4-palindrome

2

The popped element is 30

Enter your choice

1-push

2-pop

3-display

4-palindrome

2

The popped element is 20

Enter your choice

1-push

2-pop

3-display

4-palindrome2

The popped element is 10

Enter your choice

1-push

2-pop

3-display

4-palindrome

2

Stack underflow

Enter your choice

1-push

2-pop

3-display

4-palindrome

4

Enter n

123

It is not a palindrome

Enter your choice

1-push

2-pop

3-display

4-palindrome

4

Enter n

121

It is a palindrome

Enter your choice

1-push

2-pop

3-display

4-palindrome

1. **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<conio.h>

#include<ctype.h>

char stk[100];

int tos = -1;

void push(char opr)

{

stk[++tos] = opr;

}

char pop()

{

return(stk[tos--]);

}

int preced(char opr)

{

if(opr=='^'||opr=='%') return(4);

if(opr=='\*'||opr=='/') return(3);

if(opr=='+'||opr=='-') return(2);

if(opr=='('||opr=='#') return(1);

}

void main()

{

char infix[20],postfix[20];

int i,j=0;

printf("\nEnter valid INFIX expression\n");

scanf("%s",infix);

push('#');

for(i=0; infix[i]!='\0'; i++)

{

if(infix[i]=='(')

push('(');

else if(isalnum(infix[i]))

postfix[j++] = infix[i];

else if(infix[i]==')')

{

while(stk[tos] != '(')

postfix[j++] = pop();

pop();

}

else

{

while(preced(stk[tos]) >= preced(infix[i]))

postfix[j++] = pop();

push(infix[i]);

}

}

while(stk[tos] != '#')

postfix[j++] = pop();

postfix[j]='\0';

printf("\n INFIX EXPRESSION = %s",infix);

printf("\n POSTFIX EXPRESSION = %s",postfix);

getch();

}

**OUTPUT:-**

Enter valid INFIX expression

(a+b\*c)-d^e/f%g

INFIX EXPRESSION = (a+b\*c)-d^e/f%g

POSTFIX EXPRESSION = abc\*+de^fg%/-

Enter valid INFIX expression

(1+3\*6)-5%6^8

INFIX EXPRESSION = (1+3\*6)-5%6^8

POSTFIX EXPRESSION = 136\*+56%8^-

**5 Design, Develop and Implement a program in C for the following stack operations**

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

**b. Solving tower of Hanoi problem with n disks.**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#include<conio.h>

int stk[25],tos=-1;

void push(int item)

{

stk[++tos]=item;

}

int pop()

{

return (stk[tos--]);

}

int main()

{

char post[25],sym;

int op1,op2,i;

printf("Enter the postfix expression:\n");

scanf("%s",post);

for(i=0;i<strlen(post);i++)

{

sym=post[i];

switch(sym)

{

case '+':op2=pop();

op1=pop();

push(op1+op2);

break;

case '-':op2=pop();

op1=pop();

push(op1-op2);

break;

case '\*':op2=pop();

op1=pop();

push(op1\*op2);

break;

case '/':op2=pop();

op1=pop();

push(op1/op2);

break;

default:push(sym-'0');

break;

}

}

printf("The result if %d\n",pop());

}

Output1:

Enter the postfix expression:

6523+8\*+3+\*

The result if 288

Output2:

Enter the postfix expression:

93\*1-2+4\*5-

The result if 107

**Solving tower of Hanoi problem with n disks.**

#include<stdio.h>

void tower(int num,char src,char tmp,char dest)

{

if(num==1)

{

printf("Move disk 1 from peg %c to peg %c\n",src,dest);

return;

}

tower(num-1,src,dest,tmp);

printf("Move disk %d from peg %c to peg %c\n",num,src,dest);

tower(num-1,tmp,src,dest);

}

int main()

{

int num;

printf("Enter number of disks\n");

scanf("%d",&num);

tower(num,'A','B','C');

}

Output1:

Enter number of disks

3

Move disk 1 from peg A to peg C

Move disk 2 from peg A to peg B

Move disk 1 from peg C to peg B

Move disk 3 from peg A to peg C

Move disk 1 from peg B to peg A

Move disk 2 from peg B to peg C

Move disk 1 from peg A to peg C

1. **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>

#define qmax 5

char q[qmax];

int front=0,rear=-1;

void qinsert();

void qdelete();

void qdisplay();

void main()

{

int ch;

printf("\nCircular Queue operations\n");

printf("1.insert\n2.delete\n3.display\n4.exit\n");

while(1)

{

printf("Enter your choice:");

scanf("%d",&ch);

switch(ch)

{

case 1: qinsert(); break;

case 2: qdelete(); break;

case 3: qdisplay(); break;

case 4: exit(1);

default: printf("Invalid option\n");

}

}

}

void qinsert()

{

char x;

if((front==0&&rear==qmax-1)||(front>0&&rear==front-1))

printf("Queue is overflow\n");

else

{

printf("\nEnter element to be insert:");

scanf("\n%c",&x);

if(rear==qmax-1&&front>0)

{

rear=0;

q[rear]=x;

}

else

{

if((front==0&&rear==-1)||(rear!=front-1))

q[++rear]=x;

}

printf("%c is successfully inserted", x);

}

}

void qdelete()

{

char a;

if((front==0)&&(rear==-1))

{

printf("Queue is underflow\n");

return;

}

if(front==rear)

{

a=q[front];

rear=-1;

front=0;

}

else if(front==qmax-1)

{

a=q[front];

front=0;

}

else a=q[front++];

printf("Deleted element is:%c\n", a);

}

void qdisplay()

{

int i,j;

if(front==0&&rear==-1)

{

printf("Queue is underflow\n");

return;

}

if(front>rear)

{

for(i=0;i<=rear;i++)

printf("\t%c",q[i]);

for(j=front;j<=qmax-1;j++)

printf("\t%c",q[j]);

printf("\nrear is at %c\n",q[rear]);

printf("\nfront is at %c\n",q[front]);

}

else

{

for(i=front;i<=rear;i++)

printf("\t%c",q[i]);

printf("\nrear is at %c\n",q[rear]);

printf("\nfront is at %c\n",q[front]);

}

printf("\n");

}

**OUTPUT:-**

Circular Queue operations

1.insert

2.delete

3.display

4.exit

Enter your choice: 1

Enter element to be insert: A

A is successfully inserted

Enter your choice:1

Enter element to be insert: B

B is successfully inserted

Enter your choice:1

Enter element to be insert: C

C is successfully inserted

Enter your choice: 3

A B C

rear is at C

front is at A

Enter your choice: 2

Deleted element is:A

Enter your choice: B C

rear is at C

front is at B

Enter your choice: 4

1. **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 / Deletion at End of SLL**

**d. Perform Insertion / Deletion at Front of SLL(Demonstration of stack)**

**e. Exit**

#include<stdio.h>

#include<conio.h>

#include<string.h>

#include<stdlib.h>

int count=0;

struct node

{

int sem,phno;

char name[20],branch[20],usn[10];

struct node \*next;

}\*first=NULL,\*last=NULL,\*temp=NULL,\*temp1=NULL;

void create()

{

int sem,phno;

char name[20],usn[10],branch[20];

temp=(struct node \*)malloc(sizeof(struct node));

printf("Enter the details \n");

printf("\nName:"); scanf("%s",name); flushall();

printf("\nUSN:"); gets(usn); flushall();

printf("\nBranch:"); gets(branch);

printf("\nSem:"); scanf("%d",&sem);

printf("\nPhone Number:"); scanf("%d",&phno);

strcpy(temp->usn,usn);

strcpy(temp->name,name);

strcpy(temp->branch,branch);

temp->sem=sem;

temp->phno=phno;

temp->next=NULL;

count++;

}

void deletefront()

{

temp=first;

if(first==NULL)

{

printf("\n list is empty");

return;

}

if(temp->next==NULL)

{

printf("The deleted node is \n");

printf("%s\t%s\t%s\t%d\t%d",temp->name,temp->usn,temp->branch,temp->sem,temp->phno);

free(temp);

first=NULL;

}

else

{

first=temp->next;

printf("The deleted node is \n");

printf("%s\t%s\t%s\t%d\t%d",temp->name,temp->usn,temp->branch,temp->sem,temp->phno);

free(temp);

}

count--;

}

void deleteatend()

{

temp=first;

if(first==NULL)

{

printf("\n list is empty");

return;

}

if(temp->next==NULL)

{

printf("The deleted node is \n");

printf("%s\t%s\t%s\t%d\t%d",temp->name,temp->usn,temp->branch,temp->sem,temp->phno);

free(temp);

first=NULL;

}

else

{

while(temp->next!=last)

temp=temp->next;

printf("The deleted node is \n");

printf("%s\t%s\t%s\t%d\t%d",last->name,last->usn,last->branch,last->sem,last->phno);

free(last);

last=temp;

last->next=NULL;

}

count--;

}

void insertatfirst()

{

create();

if(first==NULL)

{

first=temp;

last=first;

}

else

{

temp->next=first;

first=temp;

}

}

void insertatlast()

{

create();

if(first==NULL)

{

first=temp;

last=first;

}

else

{

last->next=temp;

last=temp;

}

}

void display()

{

if(first==NULL)

{

printf("\n list is empty");

}

else

{

temp=first;

printf("The node is \n");

while(temp!=NULL)

{

printf("%s\t%s\t%s\t%d\t%d--->",temp->name,temp->usn,temp->branch,temp->sem,temp->phno);

temp=temp->next;

//printf("\n");

}

}

}

void main()

{

int ch,i,n;

clrscr();

while(1)

{

printf("\n1.Insert n details student ");

printf("\n2.Insert at beginning");

printf("\n3.Insert at last");

printf("\n4.Delete from begining");

printf("\n5.Delete from last");

printf("\n6.Display");

printf("\n7.Exit");

printf("\nEneter your choice : ");

scanf("%d",&ch);

switch(ch)

{

case 1 : printf("\nEnter the value of n ");

scanf("%d",&n);

for(i=0;i<n;i++)

insertatfirst();

break;

case 2 : insertatfirst();

break;

case 3 : insertatlast();

break;

case 4 : deletefront();

break;

case 5 : deleteatend();

break;

case 6 : display();

break;

case 7 : exit(1);

default: printf("\n Wrong Input, try again");

}

}

}

**OUTPUT:-**

1.Insert n details student

2.Insert at beginning

3.Insert at last

4.Delete from begining

5.Delete from last

6.Display

7.Exit

Eneter your choice : 1

Enter the value of n 2

Enter the details

Name: ARUNA

USN: 16IS002

Branch:ISE

Sem:3

Phone Number:1234

Enter the details

Name: VINAY

USN:16CS095

Branch:CSE

Sem:3

Phone Number:2345

1.Insert n details student

2.Insert at beginning

3.Insert at last

4.Delete from begining

5.Delete from last

6.Display

7.Exit

Eneter your choice : 6

The node is

VINAY 16CS095 CSE 3 2345--->ARUNA 16IS002 ISE 3 1234--->

1.Insert n details student

2.Insert at beginning

3.Insert at last

4.Delete from begining

5.Delete from last

6.Display

7.Exit

Eneter your choice :2

Enter the details

Name: AJAY

USN:16EC006

Branch:ECE

Sem:3

Phone Number:3456

1.Insert n details student

2.Insert at beginning

3.Insert at last

4.Delete from begining

5.Delete from last

6.Display

7.Exit

Eneter your choice : 6

The node is

AJAY 16EC006 ECE 3 3456--->VINAY 16CS095 CSE 3 2345--->ARUNA 16IS002 ISE 3 1234--->

1.Insert n details student

2.Insert at beginning

3.Insert at last

4.Delete from begining

5.Delete from last

6.Display

7.Exit

Eneter your choice :3

Enter the details

Name: ROOPA

USN:16EE030

Branch:EEE

Sem:3

Phone Number:4567

1.Insert n details student

2.Insert at beginning

3.Insert at last

4.Delete from begining

5.Delete from last

6.Display

7.Exit

Eneter your choice : 6

The node is

AJAY 16EC006 ECE 3 3456--->VINAY 16CS095 CSE 3 2345--->ARUNA 16IS002 ISE 3 1234--->ROOPA 16EE030 EEE 3 4567

1.Insert n details student

2.Insert at beginning

3.Insert at last

4.Delete from begining

5.Delete from last

6.Display

7.Exit

Eneter your choice :4

The deleted node is

AJAY 16EC006 ECE 3 3456

1.Insert n details student

2.Insert at beginning

3.Insert at last

4.Delete from begining

5.Delete from last

6.Display

7.Exit

Eneter your choice : 6

The node is

VINAY 16CS095 CSE 3 2345--->ARUNA 16IS002 ISE 3 1234--->ROOPA 16EE030 EEE 3 4567

1.Insert n details student

2.Insert at beginning

3.Insert at last

4.Delete from begining

5.Delete from last

6.Display

7.Exit

Eneter your choice : 5

The deleted node is

ROOPA 16EE030 EEE 3 4567

1.Insert n details student

2.Insert at beginning

3.Insert at last

4.Delete from begining

5.Delete from last

6.Display

7.Exit

Eneter your choice : 6

The node is

VINAY 16CS095 CSE 3 2345--->ARUNA 16IS002 ISE 3 1234--->

1. **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<conio.h>

#include<stdlib.h>

#include<string.h>

int count=0;

struct node

{

struct node \*prev;

int ssn;

long int phno;

char name[20],dept[20],desg[10];

float sal;

struct node \*next;

}\*first=NULL,\*last=NULL,\*temp=NULL,\*temp1=NULL;

void create()

{

int ssn;

long int phno;

char name[20],dept[20],desg[10];

float sal;

temp=(struct node \*)malloc(sizeof(struct node));

printf("\nEnter the employee details: ssn,name,dept,desg,sal,phno");

scanf("%d%s%s%s%f%ld",&ssn,name,dept,desg,&sal,&phno);

strcpy(temp->name,name);

strcpy(temp->dept,dept);

strcpy(temp->desg,desg);

temp->ssn=ssn;

temp->sal=sal;

temp->phno=phno;

temp->prev=NULL;

temp->next=NULL;

count++;

}

void deleteatfirst()

{

temp=first;

if(first==NULL)

{

printf("\n DLL is empty");

return;

}

if(temp->next==NULL)

{

printf("\nDeleted node is:");

printf("%d\t%s\t%s\t%s\t%f\t%ld",temp->ssn,temp->name,temp->dept,temp->desg,temp->sal,temp->phno);

free(temp);

first=NULL;

}

else

{

first=temp->next;

printf("\nDeleted node is:");

printf("%d\t%s\t%s\t%s\t%f\t%ld",temp->ssn,temp->name,temp->dept,temp->desg,temp->sal,temp->phno);

free(temp);

first->prev=NULL;

}

count--;

}

void deleteatlast()

{

temp=first;

if(first==NULL) {

printf("\n DLL is empty");

return;

}

if(temp->next==NULL)

{

printf("\nDeleted node is:");

printf("%d\t%s\t%s\t%s\t%f\t%ld",temp->ssn,temp->name,temp->dept,temp->desg,temp->sal,temp->phno);

free(temp);

first=NULL;

}

else

{

printf("\nThe deleted node is:");

temp1=last->prev;

printf("%d\t%s\t%s\t%s\t%d\t%ld",last->ssn,last->name,last->dept,last->desg,last->sal,last->phno);

free(last);

last=temp1;

last->next=NULL;

}

count--;

}

void insertatfirst()

{

create();

if(first==NULL)

{

first=temp;

last=first;

}

else

{

first->prev=temp;

temp->next=first;

first=temp;

}

}

void insertatlast()

{

create();

if(first==NULL)

{

first=temp;

last=first;

}

else

{

last->next=temp;

temp->prev=last;

last=temp;

}

}

void display()

{

if(first==NULL)

{

printf("\nDLL is empty.");

return;

}

else

{

temp=first;

printf("\n DLL is\n");

while(temp!=NULL)

{

printf("%d\t%s\t%s\t%s\t%f\t%ld\n",temp->ssn,temp->name,temp->dept,temp->desg,temp->sal,temp->phno);

temp=temp->next;

}

printf("\nThe number of nodes in the linked list is %d.",count);

}

}

void main()

{

int ch,i,n;

clrscr();

while(1)

{

printf("\nEnter the details:");

printf("\n1.Enter n employee details.");

printf("\n2.Insert at beginning.");

printf("\n3.Insert at last");

printf("\n4.Delete at beginning");

printf("\n5.Delete at last");

printf("\n6.Display");

printf("\n7.Exit");

printf("\nEnter your choice:");

scanf("%d",&ch);

switch(ch)

{

case 1:printf("Enter n value:");

scanf("%d",&n);

for(i=0;i<n;i++)

insertatlast();

break;

case 2:insertatfirst();

break;

case 3:insertatlast();

break;

case 4:deleteatfirst();

break;

case 5:deleteatlast();

break;

case 6:display();

break;

case 7:exit(1);

default:printf("Invalid input.");

}

}

}

**OUTPUT:-**

Enter the details:

1.Enter n employee details.

2.Insert at beginning.

3.Insert at last

4.Delete at beginning

5.Delete at last

6.Display

7.Exit

Enter your choice:1

Enter n value: 2

Enter the employee details:ssn,name,dept,desg,sal,phno

1 ajay cs ap 1234 1111

Enter the employee details:ssn,name,dept,desg,sal,phno

2 bhavya is ap 2345 2222

Enter the details:

1.Enter n employee details.

2.Insert at beginning.

3.Insert at last

4.Delete at beginning

5.Delete at last

6.Display

7.Exit

Enter your choice:6

DLL is

1 ajay cs ap 1234.000000 1111

2 bhavya is ap 2345.000000 2222

The number of nodes in the linked list is 2.

Enter the details:

1.Enter n employee details.

2.Insert at beginning.

3.Insert at last

4.Delete at beginning

5.Delete at last

6.Display

7.Exit

Enter your choice:2

Enter the employee details:ssn,name,dept,desg,sal,phno

3 chaitra ee ap 3456 3333

Enter the details:

1.Enter n employee details.

2.Insert at beginning.

3.Insert at last

4.Delete at beginning

5.Delete at last

6.Display

7.Exit

Enter your choice:

DLL is

3 chaitra ee ap 3456.000000 3333

1 ajay cs ap 1234.000000 1111

2 bhavya is ap 2345.000000 2222

The number of nodes in the linked list is 3.

Enter the details:

1.Enter n employee details.

2.Insert at beginning.

3.Insert at last

4.Delete at beginning

5.Delete at last

6.Display

7.Exit

Enter your choice:3

Enter the employee details:ssn,name,dept,desg,sal,phno

4 divya it ap 4567 4444

Enter the details:

1.Enter n employee details.

2.Insert at beginning.

3.Insert at last

4.Delete at beginning

5.Delete at last

6.Display

7.Exit

Enter your choice:6

DLL is

3 chaitra ee ap 3456.000000 3333

1 ajay cs ap 1234.000000 1111

2 bhavya is ap 2345.000000 2222

4 divya it ap 4567.000000 4444

The number of nodes in the linked list is 4.

Enter the details:

1.Enter n employee details.

2.Insert at beginning.

3.Insert at last

4.Delete at beginning

5.Delete at last

6.Display

7.Exit

Enter your choice:4

Deleted node is:3 chaitra ee ap 3456.000000 3333

Enter the details:

1.Enter n employee details.

2.Insert at beginning.

3.Insert at last

4.Delete at beginning

5.Delete at last

6.Display

7.Exit

Enter your choice:6

DLL is

1 ajay cs ap 1234.000000 1111

2 bhavya is ap 2345.000000 2222

4 divya it ap 4567.000000 4444

The number of nodes in the linked list is 3.

Enter the details:

1.Enter n employee details.

2.Insert at beginning.

3.Insert at last

4.Delete at beginning

5.Delete at last

6.Display

7.Exit

Enter your choice:5

The deleted node is:4 divya it ap 0 -687865856

Enter the details:

1.Enter n employee details.

2.Insert at beginning.

3.Insert at last

4.Delete at beginning

5.Delete at last

6.Display

7.Exit

Enter your choice:6

DLL is

1 ajay cs ap 1234.000000 1111

2 bhavya is ap 2345.000000 2222

The number of nodes in the linked list is 2.

Enter the details:

1.Enter n employee details.

2.Insert at beginning.

3.Insert at last

4.Delete at beginning

5.Delete at last

6.Display

7.Exit

Enter your choice:7

1. **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<alloc.h>

#include<math.h>

struct node

{

int cf, px, py, pz;

int flag;

struct node \*link;

};

typedef struct node NODE;

NODE\* getnode()

{

NODE \*x;

x=(NODE\*)malloc(sizeof(NODE));

if(x==NULL)

{

printf("Insufficient memory\n");

exit(0);

}

return x;

}

void display(NODE \*head)

{

NODE \*temp;

if(head->link==head)

{

printf("Polynomial does not exist\n");

return;

}

temp=head->link;

printf("\n");

while(temp!=head)

{

printf("%d x^%d y^%d z^%d",temp->cf,temp->px,temp->py,temp->pz);

if(temp->link != head)

printf(" + ");

temp=temp->link;

}

printf("\n");

}

NODE\* insert\_rear(int cf,int x,int y,int z,NODE \*head)

{

NODE \*temp,\*cur;

temp=getnode();

temp->cf=cf;

temp->px=x;

temp->py=y;

temp->pz=z;

temp->flag=0;

cur=head->link;

while(cur->link!=head)

{

cur=cur->link;

}

cur->link=temp;

temp->link=head;

return head;

}

NODE\* read\_poly(NODE \*head)

{

int px, py, pz, cf;

int ch;

printf("\nEnter coeff: ");

scanf("%d",&cf);

printf("\nEnter x, y, z powers(0-indiacate NO term): "); scanf("%d%d%d", &px, &py, &pz);

head=insert\_rear(cf,px,py,pz,head);

printf("\nIf you wish to continue press 1 otherwise 0: ");

scanf("%d",&ch);

while(ch!=0)

{

printf("\nEnter coeff: ");

scanf("%d",&cf);

printf("\nEnter x, y, z powers(0-indiacate NO term): ");

scanf("%d%d%d", &px, &py, &pz); head=insert\_rear(cf,px,py,pz,head);

printf("\nIf you wish to continue press 1 otherwise 0: ");

scanf("%d", &ch);

}

return head;

}

NODE\* add\_poly(NODE \*h1,NODE \*h2,NODE \*h3)

{

NODE \*p1,\*p2;

int x1,x2,y1,y2,z1,z2,cf1,cf2,cf;

p1=h1->link;

while(p1!=h1)

{

x1=p1->px;

y1=p1->py;

z1=p1->pz;

cf1=p1->cf;

p2=h2->link;

while(p2!=h2)

{

x2=p2->px;

y2=p2->py;

z2=p2->pz;

cf2=p2->cf;

if(x1==x2 && y1==y2 && z1==z2)

break;

p2=p2->link;

}

if(p2!=h2)

{

cf=cf1+cf2;

p2->flag=1;

if(cf!=0)

h3=insert\_rear(cf,x1,y1,z1,h3);

}

else

h3=insert\_rear(cf1,x1,y1,z1,h3);

p1=p1->link;

}

p2=h2->link;

while(p2!=h2)

{

if(p2->flag==0)

h3=insert\_rear(p2->cf,p2->px,p2->py,p2->pz,h3);

p2=p2->link;

}

return h3;

}

void evaluate(NODE \*he)

{

NODE \*head;

int x, y, z;

float result=0.0;

head=he;

printf("\nEnter x, y, z, terms to evaluate:\n");

scanf("%d%d%d", &x, &y, &z);

he=he->link;

while(he != head)

{

result = result + (he->cf \* pow(x,he->px) \* pow(y,he->py) \* pow(z,he->pz));

he=he->link;

}

printf("\nPolynomial result is: %f", result);

}

void main()

{

NODE \*h1,\*h2,\*h3,\*he;

int ch;

clrscr();

while(1)

{

printf("\n\n1.Evaluate polynomial\n2.Add two polynomials\n3.Exit\n");

printf("Enter your choice: ");

scanf("%d", &ch);

switch(ch)

{

case 1: he=getnode();

he->link=he;

printf("\nEnter polynomial to evaluate:\n");

he=read\_poly(he);

display(he);

evaluate(he);

free(he);

break;

case 2: h1=getnode();

h2=getnode();

h3=getnode();

h1->link=h1;

h2->link=h2;

h3->link=h3;

printf("\nEnter the first polynomial:");

h1=read\_poly(h1);

printf("\nEnter the second polynomial:");

h2=read\_poly(h2);

h3=add\_poly(h1,h2,h3);

printf("\nFirst polynomial is: ");

display(h1);

printf("\nSecond polynomial is: ");

display(h2);

printf("\nThe sum of 2 polynomials is: "); display(h3);

break;

case 3:exit(0);

break;

default:printf("\nInvalid entry");

break;

}

getch();

}

}

**OUTPUT:-**

1.Evaluate polynomial

2.Add two polynomials

3.Exit

Enter your choice: 1

Enter polynomial to evaluate:

Enter coeff: 6

Enter x, y, z powers(0-indiacate NO term):

2 2 1

If you wish to continue press 1 otherwise 0: 1

Enter coeff: -4

Enter x, y, z powers(0-indiacate NO term):

0 1 5

If you wish to continue press 1 otherwise 0: 1

Enter coeff: 3

Enter x, y, z powers(0-indiacate NO term):

3 1 1

If you wish to continue press 1 otherwise 0: 1

Enter coeff: 2

Enter x, y, z powers(0-indiacate NO term):

1 5 1

If you wish to continue press 1 otherwise 0: 1

Enter coeff: -2

Enter x, y, z powers(0-indiacate NO term):

1 1 3

If you wish to continue press 1 otherwise 0: 0

6 x^2 y^2 z^1 + -4 x^0 y^1 z^5 + 3 x^3 y^1 z^1 + 2 x^1 y^5 z^1 + -2 x^1 y^1 z^3

Enter x, y, z, terms to evaluate:

2 2 2

Polynomial result is: 224.000000

1.Evaluate polynomial

2.Add two polynomials

3.Exit

Enter your choice: 2

Enter the first polynomial:

Enter coeff: 1

Enter x, y, z powers(0-indiacate NO term):

1 1 1

If you wish to continue press 1 otherwise 0: 1

Enter coeff: 3

Enter x, y, z powers(0-indiacate NO term):

3 4 5

If you wish to continue press 1 otherwise 0: 0

Enter the second polynomial:

Enter coeff: 2

Enter x, y, z powers(0-indiacate NO term):

5 4 6

If you wish to continue press 1 otherwise 0: 1

Enter coeff: 1

Enter x, y, z powers(0-indiacate NO term):

1 1 1

If you wish to continue press 1 otherwise 0: 0

First polynomial is:

1 x^1 y^1 z^1 + 3 x^3 y^4 z^5

Second polynomial is:

2 x^5 y^4 z^6 + 1 x^1 y^1 z^1

The sum of 2 polynomials is:

2 x^1 y^1 z^1 + 3 x^3 y^4 z^5 + 2 x^5 y^4 z^6

1.Evaluate polynomial

2.Add two polynomials

3.Exit

Enter your choice: 3

1. **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 from BST**

**e. Exit**

#include <stdio.h>

#include <conio.h>

#include <stdlib.h>

struct BST

{

int item;

struct BST \*llink, \*rlink;

};

typedef struct BST\* NODE;

NODE insert(NODE);

void inorder(NODE);

void preorder(NODE);

void postorder(NODE);

NODE search(NODE, int);

NODE Delete(NODE, int);

void main()

{

int choice, key,n,i;

NODE root = NULL, tmp, parent;

clrscr();

while(1)

{

printf("\n1.Create");

printf("\n2.Traverse the Tree in Preorder, Inorder, Postorder");

printf("\n3.Search");

printf("\n4.Delete an element from the Tree");

printf("\n5.Exit");

printf("\nEnter your choice :");

scanf("%d", &choice);

switch (choice)

{

case 1: printf("\n enter the number of nodes");

scanf("%d",&n);

for(i=0;i<n;i++)

root = insert(root);

break;

case 2:if (root == NULL)

printf("Tree Is Not Created");

else

{

printf("\nThe Inorder display : ");

inorder(root);

printf("\nThe Preorder display : ");

preorder(root);

printf("\nThe Postorder display : ");

postorder(root);

}

break;

case 3:printf("\nEnter Element to be searched :");

scanf("%d", &key);

tmp = search(root, key);

if(tmp == NULL)

printf("Element does not exists\n");

else

printf("\nThe element %d found", tmp->item);

break;

case 4: printf("\nEnter Element to be deleted :");

scanf("%d", &key);

root = Delete(root, key);

break;

default: exit(0);

}

}

}

/\* This function is for creating a binary search tree \*/

NODE insert(NODE root)

{

NODE temp, cur, prev;

int item;

printf("\nEnter The Element ");

scanf("%d", &item);

temp = (NODE) malloc(sizeof(struct BST));

temp->llink = NULL;

temp->rlink = NULL;

temp->item = item;

if (root == NULL)

return temp;

prev = NULL;

cur = root;

while(cur != NULL)

{

prev = cur;

if (item < cur-> item)

cur = cur->llink;

else

cur = cur->rlink;

}

if (item < prev->item)

prev->llink = temp;

else

prev->rlink = temp;

return root;

}

/\* This function displays the tree in inorder fashion \*/

void inorder(NODE root)

{

if (root != NULL)

{

inorder(root->llink);

printf("%d\t", root->item);

inorder(root->rlink);

}

}

/\* This function displays the tree in preorder fashion \*/

void preorder(NODE root)

{

if (root != NULL)

{

printf("%d\t", root->item);

preorder(root->llink);

preorder(root->rlink);

}

}

/\* This function displays the tree in postorder fashion \*/

void postorder(NODE root)

{

if (root != NULL)

{

postorder(root->llink);

postorder(root->rlink);

printf("%d\t", root->item);

}

}

NODE search(NODE root, int key)

{

NODE cur;

if(root == NULL)

return NULL;

cur = root;

while(cur != NULL)

{

if(key == cur->item)

return cur;

if(key<cur->item)

cur = cur->llink;

else

cur = cur->rlink;

}

return NULL;

}

NODE Delete(NODE root, int data)

{

NODE temp;

int min;

if (root == NULL)

{

return NULL;

}

if (data < root->item)

{ // data is in the left sub tree.

root->llink = Delete(root->llink, data);

}

else if (data > root->item)

{ // data is in the right sub tree.

root->rlink = Delete(root->rlink, data);

}

else

{

// case 1: no children

if (root->llink == NULL && root->rlink == NULL)

{

free(root); // wipe out the memory, in C, use free function

root = NULL;

}

// case 2: one child (right)

else if (root->llink == NULL)

{

temp = root; // save current node as a backup

root = root->rlink;

free(temp);

}

// case 3: one child (left)

else if (root->rlink == NULL)

{

temp = root; // save current node as a backup

root = root->llink;

free(temp);

}

// case 4: two children

else

{

min = FindMin(root->rlink); // find minimal value of right sub tree

root->item = min; // duplicate the node

root->rlink = Delete(root->rlink, min); // delete the duplicate node

}

}

return root; // parent node can update reference

}

int FindMin(NODE root) {

if (root == NULL) {

return -1; // or undefined.

}

if (root->llink != NULL) {

return FindMin(root->llink); // left tree is smaller

}

return root->item;

}

**OUTPUT:-**

1.Create

2.Traverse the Tree in Preorder, Inorder, Postorder

3.Search

4.Delete an element from the Tree

5.Exit

Enter your choice :1

enter the number of nodes 7

Enter The Element 16

Enter The Element 9

Enter The Element 11

Enter The Element 24

Enter The Element 27

Enter The Element 4

Enter The Element 17

1.Create

2.Traverse the Tree in Preorder, Inorder, Postorder

3.Search

4.Delete an element from the Tree

5.Exit

Enter your choice :2

The Inorder display : 4 9 11 16 17 24 27

The Preorder display : 16 9 4 11 24 17 27

The Postorder display : 4 11 9 17 27 24 16

1.Create

2.Traverse the Tree in Preorder, Inorder, Postorder

3.Search

4.Delete an element from the Tree

5.Exit

Enter your choice :3

Enter Element to be searched :11

The element 11 found

1.Create

2.Traverse the Tree in Preorder, Inorder, Postorder

3.Search

4.Delete an element from the Tree

5.Exit

Enter your choice :3

Enter Element to be searched :22

Element does not exists

1.Create

2.Traverse the Tree in Preorder, Inorder, Postorder

3.Search

4.Delete an element from the Tree

5.Exit

Enter your choice :4

Enter Element to be deleted :16

1.Create

2.Traverse the Tree in Preorder, Inorder, Postorder

3.Search

4.Delete an element from the Tree

5.Exit

Enter your choice : 2

The Inorder display : 4 9 11 17 24 27

The Preorder display : 17 9 4 11 24 27

The Postorder display : 4 11 9 27 24 17

1.Create

2.Traverse the Tree in Preorder, Inorder, Postorder

3.Search

4.Delete an element from the Tree

5.Exit

Enter your choice :5

1. **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 DFS/BFS method**

#include<stdio.h>

#include<stdlib.h>

int n,a[10][10],i,j,source,s[10],reach[10],choice,count=0;

void bfs(int n,int a[10][10],int source,int s[]) //BFS Algorithm

{

int q[10],u;

int front=1,rear=1;

s[source]=1;

q[rear]=source;

while(front<=rear)

{

u=q[front];

front=front+1;

for(i=1;i<=n;i++)

if(a[u][i]==1 &&s[i]==0)

{ rear=rear+1;

q[rear]=i;

s[i]=1;

}

}

}

void dfs(int v) //DFS Algorithm

{

int i;

reach[v]=1;

for(i=1;i<=n;i++)

{

if(a[v][i] && !reach[i])

{

printf("\n %d->%d",v,i);

count++;

dfs(i);

}

}

}

int main()

{

clrscr();

printf("Enter the number of nodes : ");

scanf("%d",&n);

printf("\n Enter the adjacency matrix\n");

for(i=1;i<=n;i++) //Provide matrix of 0’s and 1’s

for(j=1;j<=n;j++)

scanf("%d",&a[i][j]);

while(1)

{

printf("\nEnter your choice\n");

printf("1.BFS\n 2.DFS\n 3.Exit\n");

scanf("%d",&choice);

switch(choice)

{

case 1: printf("\n Enter the source :");

scanf("%d",&source); //Provide source for BFS

for(i=1;i<=n;i++)

s[i]=0;

bfs(n,a,source,s);

for(i=1;i<=n;i++)

{

if(s[i]==0)

printf("\n The node %d is not reachable",i);

else

printf("\n The node %d is reachable",i);

}

break;

case 2: dfs(1);

if(count==n-1)

printf("\nThe graph is connected.");

else

printf("\nThe graph is not connected.");

break;

case 3: exit(0);

}

}

}

**OUTPUT:-**

Enter the number of nodes : 5

Enter the adjacency matrix

0 1 0 1 0

0 0 0 1 0

0 1 0 0 0

0 0 0 0 1

0 0 1 0 0

Enter your choice

1.BFS

2.DFS

3.Exit

1

Enter the source :2

The node 1 is not reachable

The node 2 is reachable

The node 3 is reachable

The node 4 is reachable

The node 5 is reachable

Enter your choice

1.BFS

2.DFS

3.Exit

2

1->2

2->4

4->5

5->3

The graph is connected.

Enter your choice

1.BFS

2.DFS

3.Exit

3

1. **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 functionH: 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>

FILE \*fp;

struct employee

{

char name[20];

int key,salary;

}emp[20];

int n,m;

int \*ht,index;

int count = 0;

void insert(int key)

{

index = key % m;

while(ht[index] != -1)

{

printf("\ncollision detected for %d and resloved using linear probing",key);

index = (index+1)%m;

}

ht[index] = key;

count++;

}

void display()

{

int i;

if(count == 0)

{

printf("\nHash Table is empty");

return;

}

printf("\nHash Table contents are:\n ");

for(i=0; i<m; i++)

printf("\n T[%d] --> %d ", i, ht[i]);

}

void main()

{

int i;

clrscr();

printf("\nEnter the number of employee records (N) : ");

scanf("%d", &n);

printf("\nEnter the two digit memory locations (m) for hash table: ");

scanf("%d", &m);

ht = (int \*)malloc(m\*sizeof(int));

for(i=0; i<m; i++)

ht[i] = -1;

fp=fopen("C:\\PROGRAM1.txt","w");

printf("\nEnter the four digit key,name,salary values (K) for N Employee Records:\n ");

for(i=0; i<n; i++)

{

scanf("%d%s%d",&emp[i].key,&emp[i].name,&emp[i].salary);

fprintf(fp,"%d\t%s\t%d\n",emp[i].key,emp[i].name,emp[i].salary);

}

fclose(fp);

fp=fopen("C:\\PROGRAM1.txt","r");

for(i=0;i<n;i++)

{

if(count == m)

{

printf("\n~~~Hash table is full. Cannot insert the record %d key~~~",i+1);

break;

}

fscanf(fp,"%d",&emp[i].key);

insert(emp[i].key);

}

fclose(fp);

//Displaying Keys inserted into hash table

display();

getch();

}

**OUTPUT:-**

Enter the number of employee records (N) : 5

Enter the two digit memory locations (m) for hash table: 10

Enter the four digit key,name,salary values (K) for N Employee Records:

1111 ajay 12345

7689 bhavya 12323

2222 chaitra 13212

2341 divya 14231

2342 girija 15231

collision detected for 2341 and resloved using linear probing

collision detected for 2341 and resloved using linear probing

collision detected for 2342 and resloved using linear probing

collision detected for 2342 and resloved using linear probing

Hash Table contents are:

T[0] --> -1

T[1] --> 1111

T[2] --> 2222

T[3] --> 2341

T[4] --> 2342

T[5] --> -1

T[6] --> -1

T[7] --> -1

T[8] --> -1

T[9] --> 7689