**Qn1:- Write a program in c to display the second largest element in an array using a user define function?**

**Name:- Nikit Singh Bisht**

**Roll no:- 39**

**Section:- D2**

**Branch:- B-tech(CSE)**

**#include<stdio.h>**

**#include<limits.h>**

**#define Max\_Size 40**

**int secondLargest(int \*arr,int n){**

**if(n<2)**

**return -1;**

**int firstMaxi=INT\_MIN;**

**int secondMaxi=INT\_MIN;**

**//finding 1st largest element**

**for(int i=0;i<n;i++){**

**if(arr[i]>firstMaxi){**

**firstMaxi=arr[i];**

**}**

**}**

**//finding 2nd largest element**

**for(int i=0;i<n;i++){**

**if(arr[i]>secondMaxi && arr[i]!=firstMaxi){**

**secondMaxi=arr[i];**

**}**

**}**

**return secondMaxi;**

**}**

**int main(){**

**int n;**

**printf("Enter the size of array:- ");**

**scanf("%d",&n);**

**int arr[Max\_Size];**

**printf("Enter array Element:- ");**

**for(int i=0;i<n;i++){**

**scanf("%d",&arr[i]);**

**}**

**printf("\nSecond Largest Element in array is:- %d",secondLargest(arr,n));**

**return 0;**

**}**

**1 Input:-**

**Enter the size of array:- 5**

**Enter array Element:- 1 2 6 9 4**

**Output:**

**Second Largest Element in array is:- 6**

**2 Input:-**

**Enter the size of array:- 8**

**Enter array Element:- 7 9 4 58 7 1 6 0**

**Output:**

**Second Largest Element in array is:- 9**

**3 Input:-**

**Enter the size of array:- 5**

**Enter array Element:- -9 -8 -7 -10 5**

**Output:**

**Second Largest Element in array is:- -7**

**Qn2:- Write a program in C to find array representation of sparse Matrices using a user defined function. Sparse matrix means Storing the non-zero elements of a matrix in a triple form-(row, column, value) ?**

**Name:- Nikit Singh Bisht**

**Roll no:- 39**

**Section:- D2**

**Branch:- B-tech(CSE)**

**#include<stdio.h>**

**#define Max\_Size 10**

**int k=0;**

**//Sparse Matrix**

**void sparseMatrix(int arr[][Max\_Size],int n,int m, int ans[][Max\_Size]){**

**for(int i=0;i<n;i++){**

**for(int j=0;j<m;j++){**

**if(arr[i][j]!=0){**

**ans[0][k]=i;**

**ans[1][k]=j;**

**ans[2][k++]=arr[i][j];**

**}**

**}**

**}**

**}**

**//Display Sparse Matrix**

**void displaySparseMatrix(int arr[][Max\_Size]){**

**for(int i=0;i<3;i++){**

**for(int j=0;j<k;j++){**

**printf("%d ",arr[i][j]);**

**}**

**printf("\n");**

**}**

**}**

**//Display normal Matrix**

**void displayMatrix(int arr[][Max\_Size],int n,int m){**

**for(int i=0;i<n;i++){**

**for(int j=0;j<m;j++){**

**printf("%d ",arr[i][j]);**

**}**

**printf("\n");**

**}**

**}**

**int main(){**

**int n,m;**

**int arr[Max\_Size][Max\_Size];**

**int ans[Max\_Size][Max\_Size];**

**printf("Enter the no of rows:- ");**

**scanf("%d",&n);**

**printf("Enter the no of column:- ");**

**scanf("%d",&m);**

**printf("Enter the Matrix Element:- \n");**

**for(int i=0;i<n;i++){**

**for(int j=0;j<m;j++){**

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

**}**

**}**

**printf("Input Matrix:- \n");**

**displayMatrix(arr,n,m);**

**sparseMatrix(arr,n,m,ans);**

**printf("Sparse Matrix:- \n");**

**displaySparseMatrix(ans);**

**return 0;**

**}**

**1.Input:-**

**Enter the no of rows:- 4**

**Enter the no of column:- 4**

**Enter the Matrix Element:-**

**1 0 2 0 0 0 7 4 0 0 0 0 2 3 0 0**

**Output:**

**Input Matrix:-**

**1 0 2 0**

**0 0 7 4**

**0 0 0 0**

**2 3 0 0**

**Sparse Matrix:-**

**0 0 1 1 3 3**

**0 2 2 3 0 1**

**1 2 7 4 2 3**

**2.Input:-**

**Enter the no of rows:- 3**

**Enter the no of column:- 4**

**Enter the Matrix Element:-**

**0 1 6 5 0 2 1 0 3 0 9 0**

**Input Matrix:-**

**0 1 6 5**

**0 2 1 0**

**3 0 9 0**

**Sparse Matrix:-**

**0 0 0 1 1 2 2**

**1 2 3 1 2 0 2**

**1 6 5 2 1 3 9**

**3.Input:-**

**Enter the no of rows:- 2**

**Enter the no of column:- 3**

**Enter the Matrix Element:-**

**1 3 0 4 0 0**

**Output:**

**Input Matrix:-**

**1 3 0**

**4 0 0**

**Sparse Matrix:-**

**0 0 1**

**0 1 0**

**1 3 4**

**Q3:- Write a program in C to implement tower of Hanoi Problem ?**

**Name:- Nikit Singh Bisht**

**Roll no:- 39**

**Section:- D2**

**Branch:- B-tech(CSE)**

**#include<stdio.h>**

**void towerOfHanoi(char S, char D, char A, int n){**

**//base case**

**if(n==1){**

**printf("\n%c -> %c",S,D);**

**return ;**

**}**

**towerOfHanoi(S,A,D,n-1);**

**printf("\n%c -> %c",S,D);**

**towerOfHanoi(A,D,S,n-1);**

**}**

**int main(){**

**int n;**

**printf("Enter the no of disk:- ");**

**scanf("%d",&n);**

**printf("\nSteps to Solve tower of Hanoi:- ");**

**towerOfHanoi('A','C','B',n);**

**return 0;**

**}**

**1.Input:-**

**Enter the no of disk:- 3**

**Output:**

**Steps to Solve tower of Hanoi:-**

**A -> C**

**A -> B**

**C -> B**

**A -> C**

**B -> A**

**B -> C**

**A -> C**

**2.Input:-**

**Enter the no of disk:- 2**

**Output:**

**Steps to Solve tower of Hanoi:-**

**A -> B**

**A -> C**

**B -> C**

**3.Input:-**

**Enter the no of disk:- 4**

**Output:**

**Steps to Solve tower of Hanoi:-**

**A -> B**

**A -> C**

**B -> C**

**A -> B**

**C -> A**

**C -> B**

**A -> B**

**A -> C**

**B -> C**

**B -> A**

**C -> A**

**B -> C**

**A -> B**

**A -> C**

**B -> C**

**Qn4:- Write a program in C to print the reverse of a string using stack?**

**Name:- Nikit Singh Bisht**

**Roll no:- 39**

**Section:- D2**

**Branch:- B-tech(CSE)**

**#include<stdio.h>**

**#define Max\_Size 40**

**void push(char ch, int\*top, char \*stack){**

**if(\*top==Max\_Size-1){**

**printf("Stack Overflow");**

**return ;**

**}**

**stack[++(\*top)]=ch;**

**}**

**char pop(char \*stack,int \*top){**

**if(\*top==-1){**

**printf("Stack UnderFlow");**

**}**

**return stack[(\*top)--];**

**}**

**int main(){**

**char str[Max\_Size];**

**char stack[Max\_Size];**

**int top=-1;**

**int i=0;**

**printf("Enter the String:- \n");**

**scanf("%[^\n]s",str);**

**while(str[i]!='\0'){**

**push(str[i],&top,stack);**

**i++;**

**}**

**i=0;**

**while(top!=-1){**

**str[i++]=pop(stack,&top);**

**}**

**printf("\nReverse String:- \n");**

**puts(str);**

**return 0;**

**}**

**1.Input:-**

**Enter the String:-**

**Graphic Era Hill University**

**Output:**

**Reverse String:-**

**ytisrevinU lliH arE cihparG**

**2.Input:-**

**Enter the String:-**

**Hello Word**

**Output:**

**Reverse String:-**

**droW olleH**

**3.Input:-**

**Enter the String:-**

**Uttarakhand**

**Output:**

**Reverse String:-**

**dnahkarattU**

**Qn5:- Write a program in C to convert an infix expression to a postfix Expression using the concept of stack?**

**Name:- Nikit Singh Bisht**

**Roll no:- 39**

**Section:- D2**

**Branch:- B-tech(CSE)**

**#include<stdio.h>**

**#include<stdlib.h>**

**#include<stdbool.h>**

**#define Max\_Size 40**

**char infixExpression[Max\_Size];**

**char stack[Max\_Size];**

**char postfixExpression[Max\_Size];**

**int top=-1;**

**//Operator push into Stack**

**void push(char ch){**

**if(top==Max\_Size-1){**

**printf("Stack OverFlow\n");**

**exit(0);**

**}**

**stack[++top]=ch;**

**}**

**//Remove Operator from Stack**

**char pop(){**

**if(top==-1){**

**printf("Stack UnderFlow\n");**

**exit(0);**

**}**

**return stack[top--];**

**}**

**// return peek element from stack**

**char peek(){**

**return stack[top];**

**}**

**//check its operand or not**

**bool isOprand(char ch){**

**if(ch>='a'&&ch<='z'||ch>='A'&&ch<='Z'||ch>='0'&&ch<='9'){**

**return true;**

**}**

**return false;**

**}**

**//precedence of operator**

**int precedence (char ch){**

**switch (ch){**

**case'\*':**

**case'/':**

**return 2;**

**case '+':**

**case '-':**

**return 1;**

**case '^':**

**return 3;**

**}**

**return -1;**

**}**

**//Convert Infix to prefix**

**void infix\_to\_postfix(){**

**int iIndex=0;**

**int pIndex=0;**

**while(infixExpression[iIndex]!='\0'){**

**if(isOprand(infixExpression[iIndex])){**

**postfixExpression[pIndex++]=infixExpression[iIndex];**

**}**

**else if(infixExpression[iIndex]=='('){**

**push(infixExpression[iIndex]);**

**}**

**else if(infixExpression[iIndex]==')'){**

**while(peek()!='('){**

**postfixExpression[pIndex++]=pop();**

**}**

**pop();**

**}**

**else{**

**if(precedence (peek())>=precedence (infixExpression[iIndex])){**

**postfixExpression[pIndex++]=pop();**

**}**

**push(infixExpression[iIndex]);**

**}**

**iIndex++;**

**}**

**while(top!=-1){**

**postfixExpression[pIndex++]=pop();**

**}**

**}**

**int main(){**

**printf("Enter the infixExpression:- \n");**

**scanf("%[^\n]s",infixExpression);**

**infix\_to\_postfix();**

**printf("Postfix Expression:- \n");**

**puts(postfixExpression);**

**return 0;**

**}**

**1.Input:-**

**Enter the infixExpression:-**

**5\*(13-4\*3)/6**

**Output:**

**Postfix Expression:-**

**51343\*-\*6/**

**2.Input:-**

**Enter the infixExpression:-**

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

**Output:**

**Postfix Expression:-**

**ab+cd-/ef\*gh+/+**

**3.Input:-**

**Enter the infixExpression:-**

**5\*(6+2)-(10/4)**

**Output:**

**Postfix Expression:-**

**562+\*104/-**

**Qn6:- Write a C program to evaluate a postfix expression and find out the result of the expression?**

**Name:- Nikit Singh Bisht**

**Roll no:- 39**

**Section:- D2**

**Branch:- B-tech(CSE)**

**#include<stdio.h>**

**#include<stdlib.h>**

**#include<string.h>**

**#include<stdbool.h>**

**#include<math.h>**

**#define Max 30**

**int top=-1;**

**int stack[Max];**

**char str[Max];**

**int size;**

**void push(int data){**

**if(top==size-1){**

**printf("Stack Overflow");**

**// exit(0);**

**}**

**else{**

**stack[++top]=data;**

**}**

**}**

**int pop(){**

**if(top==-1){**

**printf("Stack Underflow");**

**// exit(0);**

**}**

**else{**

**return stack[top--];**

**}**

**}**

**bool isDigit(char ch){**

**if(ch>='0'&&ch<='9'){**

**return true;**

**}**

**return false;**

**}**

**bool isOperator(char ch){**

**if(ch=='+'||ch=='-'||ch=='\*'||ch=='/'||ch=='^'){**

**return true;**

**}**

**return false;**

**}**

**int prefixEvaluation(){**

**int curIndex=0;**

**int operand;**

**while (str[curIndex]!='\0'){**

**if(isDigit(str[curIndex])){**

**operand=0;**

**while(isDigit(str[curIndex])){**

**operand=operand\*10+(str[curIndex]-'0');**

**curIndex++;**

**}**

**push(operand);**

**}**

**else if (isOperator(str[curIndex])){**

**int op2=pop();**

**int op1=pop();**

**switch (str[curIndex]){**

**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;**

**default:**

**break;**

**}**

**curIndex++;**

**}**

**else{**

**curIndex++;**

**}**

**}**

**return pop();**

**}**

**int main(){**

**printf("Enter the Postfix Expression:- ");**

**scanf("*%*[^\n]",str);**

**size=strlen(str);**

**printf("Prefix Evaluation is :- %d",prefixEvaluation());**

**return 0;**

**}**

**1.Input:-**

**Enter the Postfix Expression:- 5 6 2 + \* 12 4 / -**

**Output:**

**Prefix Evaluation is :- 37**

**2.Input:-**

**Enter the Postfix Expression:- 10 5 + 18 12 - \***

**Output:**

**Prefix Evaluation is :- 90**

**3.Input:-**

**Enter the Postfix Expression:- 100 50 + 2 \* 200 -**

**Output:**

**Prefix Evaluation is :- 100**

**Qn7:- Write a program in C to create a circular queue with N elements. Execute and display the results of functions enqueue() and dequeue() on the circular queue?**

**Name:- Nikit Singh Bisht**

**Roll no:- 39**

**Section:- D2**

**Branch:- B-tech(CSE)**

**#include<stdio.h>**

**#define Max 10**

**int queue[Max];**

**int size;**

**int front=-1,rear=-1;**

**void enqueue(){**

**if(front==-1){**

**front = rear = 0;**

**scanf("%d",&queue[rear]);**

**}**

**else if(front%size==(rear+1)%size){**

**printf("Queue is Full");**

**}**

**else{**

**rear=(rear+1)%size;**

**scanf("%d",&queue[rear]);**

**}**

**}**

**void dequeue(){**

**if(front==-1){**

**printf("Queue is Empty");**

**}**

**else if(front%size==rear%size){**

**front=rear=-1;**

**}**

**else{**

**front=(front+1)%size;**

**}**

**}**

**void display(){**

**printf("Queue Elements are:- \n");**

**int i=front;**

**while(i!=rear){**

**printf(" %d",queue[i]);**

**i=(i+1)%size;**

**}**

**if(i==rear){**

**printf(" %d",queue[i]);**

**}**

**}**

**int main(){**

**int m,j,k;**

**printf("Enter the size of Queue:- ");**

**scanf("%d",&size);**

**printf("Enter the Element in queue:- ");**

**for(int i=1;i<=size;i++){**

**enqueue();**

**}**

**display();**

**printf("\n Enter the No of Element you want to delete:- ");**

**scanf("%d",&m);**

**for(int i=0;i<m;i++){**

**dequeue();**

**}**

**display();**

**printf("\nEnter the no of Element you want to add:-");**

**scanf("%d",&j);**

**for(int i=0;i<j;i++){**

**enqueue();**

**}**

**display();**

**return 0;**

**}**

**1.Input:-**

**Enter the size of Queue:- 5**

**Enter the Element in queue:- 1 2 3 4 5**

**Queue Elements are:-**

**1 2 3 4 5**

**Enter the No of Element you want to delete:- 3**

**Queue Elements are:-**

**4 5**

**Enter the no of Element you want to add:- 3**

**1 2 3**

**Output:-**

**Queue Elements are:-**

**4 5 1 2 3**

**2.Input:-**

**Enter the size of Queue:- 8**

**Enter the Element in queue:- 9 8 6 2 1 4 5 7**

**Queue Elements are:-**

**9 8 6 2 1 4 5 7**

**Enter the No of Element you want to delete:- 5**

**Queue Elements are:-**

**4 5 7**

**Enter the no of Element you want to add:- 1**

**3**

**Output:**

**Queue Elements are:-**

**4 5 7 3**

**3.Input:-**

**Enter the size of Queue:- 4**

**Enter the Element in queue:- 10 20 30 40**

**Queue Elements are:-**

**10 20 30 40**

**Enter the No of Element you want to delete:- 4**

**Queue Elements are:-**

**0**

**Enter the no of Element you want to add:-2**

**10 60**

**Output:-**

**Queue Elements are:-**

**10 60**

**Qn8:- Write a program in C to implement a queue using stack?**

**Name:- Nikit Singh Bisht**

**Roll no:- 39**

**Section:- D2**

**Branch:- B-tech(CSE)**

**#include<stdio.h>**

**#include<stdlib.h>**

**#define Max\_Size 10**

**int stack1[Max\_Size];**

**int stack2[Max\_Size];**

**int top1=-1,top2=-1;**

**int size;**

**//Return Element from Stack-1**

**int pop1(){**

**if(top1==-1){**

**printf("Queue is empty");**

**exit(1);**

**}**

**return stack1[top1--];**

**}**

**//Push Element in Stack-2**

**void push2(int data){**

**if(top2==size-1){**

**printf("Queue is full");**

**return ;**

**}**

**else{**

**stack2[++top2]=data;**

**}**

**}**

**//Push Element into Stack-1**

**void push1(int data){**

**if(top1==size-1){**

**printf("Queue is full");**

**return ;**

**}**

**else{**

**stack1[++top1]=data;**

**}**

**}**

**//Return Element from Stack-2**

**int pop2(){**

**if(top2==-1){**

**printf("Queue is empty");**

**exit(1);**

**}**

**else{**

**return stack2[top2--];**

**}**

**}**

**//Add Element into Queue**

**void enqueue(int x){**

**while(top1!=-1){        //stack1 to stack2**

**int x = pop1();**

**push2(x);**

**}**

**stack1[++top1]=x;       //data in stack1**

**while (top2!=-1){**

**int x=pop2();**

**push1(x);**

**}**

**}**

**//Return top Element from Queue**

**int peek(){**

**return stack1[top1];**

**}**

**//Delete Queue Element**

**void dequeue(){**

**if(top1==-1){**

**printf("Queue is empty");**

**exit(1);**

**}**

**else{**

**top1--;**

**}**

**}**

**//Display Queue**

**void display(){**

**printf("Queue Elements are:- \n");**

**for (int  i = top1; i >=0; i--){**

**printf("%d ",stack1[i]);**

**}**

**}**

**int main(){**

**int m;**

**printf("Enter the size of queue:- ");**

**scanf("%d",&size);**

**for(int i=0;i<size;i++){**

**int a;**

**scanf("%d",&a);**

**enqueue(a);**

**}**

**printf("Enter the no of element you want to delete:- ");**

**scanf("%d",&m);**

**display();**

**printf("\nPeek Element is :- %d",peek());**

**for(int i=0;i<m;i++){**

**dequeue();**

**}**

**printf("\nAfter Deletion ");**

**display();**

**}**

**1.Input:-**

**Enter the size of queue:- 5**

**1 2 3 4 5**

**Enter the no of element you want to delete:- 4**

**Output:**

**Queue Elements are:-**

**1 2 3 4 5**

**Peek Element is :- 1**

**After Deletion Queue Elements are:-**

**5**

**2.Input:-**

**Enter the size of queue:- 8**

**7 9 5 6 2 4  2 3**

**Enter the no of element you want to delete:- 3**

**Output:**

**Queue Elements are:-**

**7 9 5 6 2 4 2 3**

**Peek Element is :- 7**

**After Deletion Queue Elements are:-**

**6 2 4 2 3**

**3.Input:-**

**Enter the size of queue:- 6**

**10 20 80 90 50 40**

**Enter the no of element you want to delete:- 1**

**Output:**

**Queue Elements are:-**

**10 20 80 90 50 40**

**Peek Element is :- 10**

**After Deletion Queue Elements are:-**

**20 80 90 50 40**

**Qn9:-Write a C program to search an element k in single linked list with n elements?**

**Name:- Nikit Singh Bisht**

**Roll no:- 39**

**Section:- D2**

**Branch:- B-tech(CSE)**

**#include<stdio.h>**

**#include<stdlib.h>**

**struct node{**

**int data;**

**struct node\*next;**

**}\*temp,\*newnode,\*head=0;**

**void display(){**

**printf("\nLinked List:- ");**

**temp=head;**

**while (temp!=NULL){**

**printf(" %d",temp->data);**

**temp=temp->next;**

**}**

**}**

**void searchElement(int element){**

**temp=head;**

**while (temp!=NULL){**

**if(temp->data==element){**

**printf("\nElement Present");**

**return ;**

**}**

**temp=temp->next;**

**}**

**printf("\nElement is not Present");**

**}**

**int main(){**

**int n;**

**printf("Enter the size of Linked List:- ");**

**scanf("%d",&n);**

**printf("Enter Linked list Element:- ");**

**for(int i=0;i<n;i++){**

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

**newnode->next=0;**

**scanf("%d",&newnode->data);**

**if(head==0){**

**head=newnode;**

**temp=newnode;**

**}**

**else{**

**temp->next=newnode;**

**temp=newnode;**

**}**

**}**

**int element;**

**printf("Enter the Element:- ");**

**scanf("%d",&element);**

**display();**

**searchElement(element);**

**return 0;**

**}**

**1.Input:-**

**Enter the size of Linked List:- 5**

**Enter Linked list Element:- 7 8 6 2 3**

**Enter the Element:- 8**

**Output:**

**Linked List:- 7 8 6 2 3**

**Element Present**

**2.Input:-**

**Enter the size of Linked List:- 6**

**Enter Linked list Element:- 1 2 3 8 5 4**

**Enter the Element:- 9**

**Output:**

**Linked List:- 1 2 3 8 5 4**

**Element is not Present**

**3.Input:-**

**Enter the size of Linked List:- 10**

**Enter Linked list Element:- 7 8 9 5 6 3 2 1 4 5**

**Enter the Element:- 6**

**Output:**

**Linked List:- 7 8 9 5 6 3 2 1 4 5**

**Element Present**

**Qn10:- Write a program in C to implement Following operation in a Doubly Linked list**

**1. Creation**

**2. Insertion**

**3. Deletion**

**Name:- Nikit Singh Bisht**

**Roll no:- 39**

**Section:- D2**

**Branch:- B-tech(CSE)**

**#include<stdio.h>**

**#include<stdlib.h>**

**struct node{**

**struct node\*pre;**

**int data;**

**struct node\*next;**

**}\*head=0,\*temp=0,\*newnode=0;**

**int count=0;**

**//Display Linked List**

**void display(){**

**printf("\nLinked List Element are:- \n");**

**temp=head;**

**while(temp!=NULL){**

**printf("%d ",temp->data);**

**temp=temp->next;**

**}**

**}**

**//Creation of Linked List**

**void creation(){**

**int n;**

**head=0;**

**count=0;**

**printf("Enter the size of linked list:- ");**

**scanf("%d",&n);**

**for (int i = 0; i < n; i++){**

**count++;**

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

**scanf("%d",&newnode->data);**

**newnode->next=0;**

**if(head==0){**

**newnode->pre=0;**

**temp=newnode;**

**head=newnode;**

**}**

**else{**

**temp->next=newnode;**

**newnode->pre=temp;**

**temp=newnode;**

**}**

**}**

**}**

**//insertion at begnning:**

**void insertBegnning(){**

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

**printf("\nEnter the Element:- ");**

**scanf("%d",&newnode->data);**

**newnode->pre=0;**

**head->pre=newnode;**

**newnode->next=head;**

**head=newnode;**

**}**

**//Insertion at end**

**void insertEnd(){**

**temp=head;**

**while (temp->next!=NULL){**

**temp=temp->next;**

**}**

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

**printf("\nEnter the Element:- ");**

**scanf("%d",&newnode->data);**

**newnode->next=0;**

**temp->next=newnode;**

**newnode->pre=temp;**

**temp=newnode;**

**}**

**//Insertion in Linked list**

**void insertion(){**

**if(head==0){**

**printf("Element not found");**

**exit(0);**

**}**

**int pos,i=1;**

**temp=head;**

**printf("Enter the position:- ");**

**scanf("%d",&pos);**

**if(pos>count+1||pos<=0){**

**printf("\nInvalid position");**

**exit(0);**

**}**

**if(pos==1){**

**count++;**

**insertBegnning();**

**return ;**

**}**

**else if(pos==count+1){**

**count++;**

**insertEnd();**

**return ;**

**}**

**else if(pos>1&&pos<count+1){**

**while(i<pos-1){**

**temp=temp->next;**

**i++;**

**}**

**}**

**count++;**

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

**printf("\nEnter the Element:- ");**

**scanf("%d",&newnode->data);**

**newnode->next=temp->next;**

**temp->next=newnode;**

**newnode->pre=temp;**

**temp->next->pre=newnode;**

**temp=newnode;**

**}**

**//Deletion at beginning**

**void deletionAtBeggning(){**

**temp=head;**

**temp=temp->next;**

**head=head->next;**

**temp->pre=0;**

**}**

**//Deletion at End**

**void deletionAtEnd(){**

**temp=head;**

**while (temp->next!=NULL){**

**temp=temp->next;**

**}**

**temp->pre->next=0;**

**}**

**//Deletion in Linked list**

**void deletion(){**

**if(head==0){**

**printf("\nElement not found:");**

**exit(0);**

**}**

**temp=head;**

**int pos,i=1;**

**printf("\nEnter the position:- ");**

**scanf("%d",&pos);**

**if(pos>count){**

**printf("\nInvalid position");**

**exit(0);**

**}**

**if(pos==1){**

**count--;**

**deletionAtBeggning();**

**return ;**

**}**

**else if(pos==count){**

**deletionAtEnd();**

**count--;**

**return ;**

**}**

**else if(pos>1&&pos<count){**

**while(i<pos-1){**

**temp=temp->next;**

**i++;**

**}**

**}**

**count--;**

**temp->next=temp->next->next;**

**temp->next->pre=temp;**

**temp=temp->next;**

**}**

**int main(){**

**int n;**

**up:**

**printf("\n\nEnter the operation:- \n");**

**printf("1. Creation:- \n");**

**printf("2. Insertion:- \n");**

**printf("3. Deletion:- \n");**

**printf("4. Exit:- \n");**

**scanf("%d",&n);**

**switch (n){**

**case 1:**

**creation();**

**break;**

**case 2:**

**insertion();**

**break;**

**case 3:**

**deletion();**

**break;**

**case 4:**

**exit(0);**

**default:**

**break;**

**}**

**display();**

**goto up;**

**return 0;**

**}**

**1.Input & Output:-**

**Enter the operation:-**

**1. Creation:-**

**2. Insertion:-**

**3. Deletion:-**

**4. Exit:-**

**1**

**Enter the size of linked list:- 5**

**1 2 3 4 5**

**Linked List Element are:-**

**1 2 3 4 5**

**Enter the operation:-**

**1. Creation:-**

**2. Insertion:-**

**3. Deletion:-**

**4. Exit:-**

**2**

**Enter the position:- 1**

**Enter the Element:- 0**

**Linked List Element are:-**

**0 1 2 3 4 5**

**Enter the operation:-**

**1. Creation:-**

**2. Insertion:-**

**3. Deletion:-**

**4. Exit:-**

**3**

**Enter the position:- 1**

**Linked List Element are:-**

**1 2 3 4 5**

**2.Input & Output:-**

**Enter the operation:-**

**1. Creation:-**

**2. Insertion:-**

**3. Deletion:-**

**4. Exit:-**

**1**

**Enter the size of linked list:- 8**

**8 9 6 5 1 2 3 7**

**Linked List Element are:-**

**8 9 6 5 1 2 3 7**

**Enter the operation:-**

**1. Creation:-**

**2. Insertion:-**

**3. Deletion:-**

**4. Exit:-**

**2**

**Enter the position:- 8**

**Enter the Element:- 10**

**Linked List Element are:-**

**8 9 6 5 1 2 3 10 7**

**Enter the operation:-**

**1. Creation:-**

**2. Insertion:-**

**3. Deletion:-**

**4. Exit:-**

**3**

**Enter the position:- 2**

**Linked List Element are:-**

**8 6 5 1 2 3 10 7**

**Qn11:- Write a program in C to implement a stack using a linked list?**

**Name:- Nikit Singh Bisht**

**Roll no:- 39**

**Section:- D2**

**Branch:- B-tech(CSE)**

**#include<stdio.h>**

**#include<stdlib.h>**

**struct node{**

**int data;**

**struct node\* next;**

**}\*top=0,\*temp,\*newnode;**

**int size;**

**int count=0;**

**void push(){**

**if(count==size){**

**printf("Stack Overflow");**

**}**

**else{**

**count++;**

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

**scanf("%d",&newnode->data);**

**newnode->next=top;**

**top=newnode;**

**}**

**}**

**void pop(){**

**if(top==NULL){**

**printf("Stack Underflow");**

**}**

**else**

**top=top->next;**

**}**

**int peek(){**

**if(top==NULL){**

**printf("Stack Underflow");**

**}**

**else{**

**return top->data;**

**}**

**}**

**void display(){**

**temp=top;**

**while(temp!=NULL){**

**printf("%d ",temp->data);**

**temp=temp->next;**

**}**

**}**

**int main(){**

**int m;**

**printf("Enter the no of element:- ");**

**scanf("%d",&size);**

**printf("Enter the element :-\n");**

**for (int i = 0; i < size; i++){**

**push();**

**}**

**printf("Enter the no of element you want delete:- ");**

**scanf("%d",&m);**

**printf("Stack Elements are:- ");**

**display();**

**printf("\nPeek Element is :- %d",peek());**

**for(int i=0;i<m;i++){**

**pop();**

**}**

**printf("\nAfter delete Stack Elements:- ");**

**display();**

**return 0;**

**}**

**1.Input:-**

**Enter the no of element:- 5**

**Enter the element :- 1 2 3 4 5**

**Enter the no of element you want delete:- 3**

**Output:**

**Stack Elements are:- 5 4 3 2 1**

**Peek Element is :- 5**

**After deletion Stack Elements are:- 2 1**

**2.Input:-**

**Enter the no of element:- 8**

**Enter the element :-**

**9 5 6 3 7 4 8 10**

**Output:**

**Enter the no of element you want delete:- 5**

**Stack Elements are:- 10 8 4 7 3 6 5 9**

**Peek Element is :- 10**

**After deletion Stack Elements are:- 6 5 9**

**3.Input:-**

**Enter the no of element:- 6**

**Enter the element :-**

**10 20 30 40 50 60**

**Enter the no of element you want delete:- 5**

**Output:**

**Stack Elements are:- 60 50 40 30 20 10**

**Peek Element is :- 60**

**After delete Stack Elements:- 10**

**Qn12:- Given two polynomial numbers represented by linked list, develop a program in C using a functions that adds these lists.(Adding means to add the coefficients who have the same variable powers.)**

**Name:- Nikit Singh Bisht**

**Roll no:- 39**

**Section:- D2**

**Branch:- B-tech(CSE)**

**#include<stdio.h>**

**#include<stdlib.h>**

**#include<stdbool.h>**

**struct node{**

**int data;**

**int coefficient;**

**struct node\*next;**

**}\*p=0,\*q=0,\*res=0,\*temp,\*newnode;**

**void createList(struct node\* \*x){**

**int choice;**

**printf("\nEnter the no of element :- ");**

**scanf("%d",&choice);**

**for (int i=0;i<choice;i++){**

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

**printf("\nEnter the data:- ");**

**scanf("%d",&newnode->data);**

**printf("Enter the coefficient:- ");**

**scanf("%d",&newnode->coefficient);**

**newnode->next=0;**

**if(\*x==0){**

**temp=newnode;**

**\*x=newnode;**

**}**

**else{**

**temp->next=newnode;**

**temp=newnode;**

**}**

**}**

**}**

**void display(struct node \*x){**

**temp=x;**

**while (temp!=NULL){**

**printf("\nData:- %d\t coefficient:- %d",temp->data,temp->coefficient);**

**temp=temp->next;**

**}**

**}**

**void add(){**

**while (p!=NULL&&q!=NULL){**

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

**newnode->next=0;**

**if(res==0){**

**res=newnode;**

**temp=newnode;**

**}**

**else{**

**temp->next=newnode;**

**temp=newnode;**

**}**

**if(p->coefficient>q->coefficient){**

**temp->data=p->data;**

**temp->coefficient=p->coefficient;**

**p=p->next;**

**}**

**else if(q->coefficient>p->coefficient){**

**temp->data=q->data;**

**temp->coefficient=q->coefficient;**

**q=q->next;**

**}**

**else{**

**temp->data=q->data+p->data;**

**temp->coefficient=q->coefficient;**

**p=p->next;**

**q=q->next;**

**}**

**}**

**while(p!=NULL){**

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

**newnode->next=0;**

**temp->next=newnode;**

**temp=newnode;**

**temp->data=p->data;**

**temp->coefficient=p->coefficient;**

**p=p->next;**

**}**

**while(q!=NULL){**

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

**newnode->next=0;**

**temp->next=newnode;**

**temp=newnode;**

**temp->data=q->data;**

**temp->coefficient=q->coefficient;**

**q=q->next;**

**}**

**}**

**int main(){**

**printf("\nEnter First polynomial data:- \n");**

**createList(&p);**

**printf("\nEnter second polynomial data:- \n");**

**createList(&q);**

**printf("\n\nfirst polynomial:- ");**

**display(p);**

**printf("\n\nsecond polynomial:-");**

**display(q);**

**add();**

**printf("\n\nAddition of two polynomial is :- ");**

**display(res);**

**return 0;**

**}**

**1.Input:-**

**Enter First polynomial data:-**

**Enter the no of element :- 3**

**Enter the data:- 3**

**Enter the coefficient:- 2**

**Enter the data:- 2**

**Enter the coefficient:- 1**

**Enter the data:- 9**

**Enter the coefficient:- 0**

**Enter second polynomial data:-**

**Enter the no of element :- 3**

**Enter the data:- 6**

**Enter the coefficient:- 3**

**Enter the data:- 5**

**Enter the coefficient:- 2**

**Enter the data:- 9**

**Enter the coefficient:- 1**

**Output:**

**First polynomial:-**

**Data:- 3         coefficient:- 2**

**Data:- 2         coefficient:- 1**

**Data:- 9         coefficient:- 0**

**Second polynomial:-**

**Data:- 6         coefficient:- 3**

**Data:- 5         coefficient:- 2**

**Data:- 9         coefficient:- 1**

**Addition of two polynomial is :-**

**Data:- 6         coefficient:- 3**

**Data:- 8         coefficient:- 2**

**Data:- 11        coefficient:- 1**

**Data:- 9         coefficient:- 0**

**2.Input:-**

**Enter First polynomial data:-**

**Enter the no of element :- 3**

**Enter the data:- 3**

**Enter the coefficient:- 2**

**Enter the data:- 2**

**Enter the coefficient:- 1**

**Enter the data:- 7**

**Enter the coefficient:- 0**

**Enter second polynomial data:-**

**Enter the no of element :- 3**

**Enter the data:- 5**

**Enter the coefficient:- 3**

**Enter the data:- 2**

**Enter the coefficient:- 2**

**Enter the data:- 1**

**Enter the coefficient:- 1**

**Output:**

**first polynomial:-**

**Data:- 3         coefficient:- 2**

**Data:- 2         coefficient:- 1**

**Data:- 7         coefficient:- 0**

**second polynomial:-**

**Data:- 5         coefficient:- 3**

**Data:- 2         coefficient:- 2**

**Data:- 1         coefficient:- 1**

**Addition of two polynomial is :-**

**Data:- 5         coefficient:- 3**

**Data:- 5         coefficient:- 2**

**Data:- 3         coefficient:- 1**

**Data:- 7         coefficient:- 0**

**3.Input:-**

**Enter First polynomial data:-**

**Enter the no of element :- 2**

**Enter the data:- 2**

**Enter the coefficient:- 5**

**Enter the data:- 3**

**Enter the coefficient:- 0**

**Enter second polynomial data:-**

**Enter the no of element :- 1**

**Enter the data:- 2**

**Enter the coefficient:- 1**

**Output:**

**first polynomial:-**

**Data:- 2         coefficient:- 5**

**Data:- 3         coefficient:- 0**

**second polynomial:-**

**Data:- 2         coefficient:- 1**

**Addition of two polynomial is :-**

**Data:- 2         coefficient:- 5**

**Data:- 2         coefficient:- 1**

**Data:- 3         coefficient:- 0**