S.Y.B.S.C(C.S)

Data Structure And Algorithm Using C

Semester III

Q1] Improved Bubble Sort For:-

A]Integer Value Input:-

Input:-

#include<stdio.h>

void input(int a[],int n);

void bubble(int a[],int n);

void print(int a[],int n);

int main() {

int n;

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

scanf("%d",&n);

int a[n];

input(a,n);

bubble(a,n);

print(a,n);

}

void input(int a[],int n) {

  int i;

  printf("Enter the numbers:-\n");

  for(i=0;i<n;i++) {

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

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

  }

}

void bubble(int a[],int n) {

  printf("Sorting The Elements!!!!\n");

  int i,j,temp;

  for(i=n-1;i>0;i--) {

    int flag=0;

    for(j=0;j<i;j++) {

if(a[j]>a[j+1]){

  flag=1;

  temp=a[j];

a[j]=a[j+1];

a[j+1]=temp;

}

    }

    if(flag != 1) {

break;

}

  }

}

void print(int a[],int n) {

  int i;

  printf("Printing The Elements After Sorting!!!!\n");

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

  printf("%d:- %d\n",i+1,a[i]);

}

}

Output:-

PS C:\Users\ADMIN\OneDrive\Desktop\Cprogramming> ./a.exe

Enter the size of the array:- 6

Enter the numbers:-

1:- 3

2:- 5

3:- 4

4:- 5

5:- 5

6:- 3

Sorting The Elements!!!!

Printing The Elements After Sorting!!!!

1:- 3

2:- 3

3:- 4

4:- 5

5:- 5

6:- 5

B]Char Value Input:-

Input:-

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

void input(char \*a[],int n);

void bubble(char \*a[],int n);

void print(char \*a[],int n);

int main() {

int n;

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

scanf("%d",&n);

char \*a[n];

input(a,n);

bubble(a,n);

print(a,n);

}

void input(char \*a[],int n) {

  int i;

  printf("Enter the numbers:-\n");

  for(i=0;i<n;i++) {

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

a[i]=(char\*)malloc(sizeof(char)\*5);

scanf("%s",a[i]);

  }

}

void bubble(char \*a[],int n) {

  printf("Sorting The Elements!!!!\n");

  int i,j;

  char temp[i];

  for(i=n-1;i>0;i--) {

    int flag=0;

    for(j=0;j<i;j++) {

if(strcmp(a[j],a[j+1])>0){

  flag=1;

  strcpy(temp,a[j]);

strcpy(a[j],a[j+1]);

strcpy(a[j+1],temp);

}

    }

    if(flag != 1) {

break;

}

  }

}

void print(char \*a[],int n) {

  int i;

  printf("Printing The Elements After Sorting!!!!\n");

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

  printf("%d:- %s\n",i+1,a[i]);

}

}

Output:-

Enter the size of the array:- 6

Enter the numbers:-

1:- ZZ

2:- XX

3:- YY

4:- AA

5:- HH

6:- NN

Sorting The Elements!!!!

Printing The Elements After Sorting!!!!

1:- AA

2:- HH

3:- NN

4:- XX

5:- YY

6:- ZZ

Q2] Insertion Sort For:-

A]Int Value Input:-

Input:-

#include<stdio.h>

void input(int a[],int n);

void Insertion(int a[],int n);

void print(int a[],int n);

int main() {

int n;

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

scanf("%d",&n);

int a[n];

input(a,n);

Insertion(a,n);

print(a,n);

}

void input(int a[],int n) {

  int i;

  printf("Enter the numbers:-\n");

  for(i=0;i<n;i++) {

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

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

  }

}

void Insertion(int a[],int n) {

  printf("Sorting The Elements!!!!\n");

  int i,j,temp;

  for(i=1;i<n;i++) {

    temp=a[i];

    for(j=i-1;j>=0;j--) {

if(a[j]>temp){

  a[j+1]=a[j];

}

else{

  break;

}

    }

    a[j+1]=temp;

  }

}

void print(int a[],int n) {

  int i;

  printf("Printing The Elements After Sorting!!!!\n");

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

  printf("%d:- %d\n",i+1,a[i]);

}

}

Output:-

Enter the size of the array:- 6

Enter the numbers:-

1:- 23

2:- 5

3:- 76

4:- 3

5:- 7

6:- 5

Sorting The Elements!!!!

Printing The Elements After Sorting!!!!

1:- 3

2:- 5

3:- 5

4:- 7

5:- 23

6:- 76

B]Char Value Input:-

Input:-

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

void input(char \*a[],int n);

void Insertion(char \*a[],int n);

void print(char \*a[],int n);

int main() {

int n;

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

scanf("%d",&n);

char \*a[n];

input(a,n);

Insertion(a,n);

print(a,n);

}

void input(char \*a[],int n) {

  int i;

  printf("Enter the numbers:-\n");

  for(i=0;i<n;i++) {

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

a[i]=(char \*)malloc(sizeof(char)\*5);

scanf("%s",a[i]);

  }

}

void Insertion(char \*a[],int n) {

  printf("Sorting The Elements!!!!\n");

  int i,j;

  char temp[5];

  for(i=1;i<n;i++) {

    strcpy(temp,a[i]);

    for(j=i-1;j>=0;j--) {

if(strcmp(a[j],temp)>0){

  strcpy(a[j+1],a[j]);

}

else{

  break;

}

    }

    strcpy(a[j+1],temp);

  }

}

void print(char \*a[],int n) {

  int i;

  printf("Printing The Elements After Sorting!!!!\n");

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

  printf("%d:- %s\n",i+1,a[i]);

}

}

Output:-

Enter the size of the array:- 6

Enter the numbers:-

1:- KK

2:- D

3:- SS

4:- HG

5:- BB

6:- AA

Sorting The Elements!!!!

Printing The Elements After Sorting!!!!

1:- AA

2:- BB

3:- D

4:- HG

5:- KK

6:- SS

Q3]Merge Sort:-

A]Int Value Input:-

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

void input(int a[],int n);

void merge(int a[],int lb,int mid,int ub);

void merge\_sort(int a[],int lb,int ub);

void print(int a[],int n);

int main() {

int n;

printf("Enter The Size Of The Array:- ");

scanf("%d",&n);

int a[n];

input(a,n);

merge\_sort(a,0,n-1);

print(a,n);

return 0;

}

void input(int a[],int n){

  printf("Enter The Numbers:- \n");

  for(int i=0;i<n;i++) {

    printf("%d] ",i+1);

        scanf("%d",&a[i]);

  }

}

void merge(int a[],int lb,int mid,int ub){

int i,j,k;

int temp[50];

i=lb;

j=mid+1;

k=lb;

while(i<=mid && j<=ub){

  if(a[i]<a[j]){

    temp[k]=a[i];

    i++;

    k++;

  }

  else{

    temp[k]=a[j];

    j++;

    k++;

  }

}

  while(i<=mid) {

    temp[k]=a[i];

    i++;

    k++;

  }

  while(j<=ub) {

    temp[k]=a[j];

    j++;

    k++;

  }

  for(k=lb;k<=ub;k++) {

    a[k]=temp[k];

  }

}

void print(int a[],int n) {

  printf("Printing The Sorted Array:- \n");

  for(int i=0;i<n;i++) {

    printf("%d] %d\n",i+1,a[i]);

  }

}

void merge\_sort(int a[],int lb,int ub) {

  int mid;

  if(lb<ub){

  mid=(lb+ub)/2;

merge\_sort(a,lb,mid);

merge\_sort(a,mid+1,ub);

merge(a,lb,mid,ub);

  }

}

Output:-

Enter The Size Of The Array:- 5

Enter The Numbers:-

1] 4

2] 9

3] 3

4] 7

5] 5

Printing The Sorted Array:-

1] 3

2] 4

3] 5

4] 7

5] 9

B]Char Value Input:-

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

void input(char \*a[],int n);

void print(char \*a[],int n);

void input(char \*a[],int n){

  printf("Enter The Numbers:- \n");

  for(int i=0;i<n;i++) {

    printf("%d] ",i+1);

    a[i]=(char\*)malloc(sizeof(char)\*10);

        scanf("%s",a[i]);

  }

}

void print(char \*a[],int n) {

  printf("Printing The Sorted Array:- \n");

  for(int i=0;i<n;i++) {

    printf("%d] %s\n",i+1,a[i]);

  }

}

void merge(char \*a[],int lb,int mid,int ub);

void merge\_sort(char \*a[],int lb,int ub);

int main() {

int n;

printf("Enter The Size Of The Array:- ");

scanf("%d",&n);

char \*a[n];

input(a,n);

merge\_sort(a,0,n-1);

print(a,n);

return 0;

}

void merge(char \*a[],int lb,int mid,int ub) {

int i,j,k;

i=lb;

j=mid+1;

k=0;

int l;

char \*temp[50];

for(l=lb;l<=ub;l++){

temp[l]=(char\*)malloc(sizeof(char)\*(ub-lb+1));

}

while(i<=mid && j<=ub){

    if(strcmp(a[i],a[j])<=0){

        temp[k]=a[i];

        i++;

        k++;

    }

    else {

        temp[k]=a[j];

        j++;

        k++;

    }

}

while(i<=mid){

    temp[k]=a[i];

    i++;

    k++;

}

while(j<=ub) {

    temp[k]=a[j];

    j++;

    k++;

}

for(i=lb,k=0;i<=ub;i++,k++) {

    a[i]=temp[k];

}

}

void merge\_sort(char \*a[],int lb,int ub) {

    if(lb<ub){

   int mid;

   mid=(lb+ub)/2;

   merge\_sort(a,lb,mid);

   merge\_sort(a,mid+1,ub);

   merge(a,lb,mid,ub);

    }

}

Output:-

Enter The Size Of The Array:- 6

Enter The Numbers:-

1] n

2] v

3] k

4] a

5] c

6] b

Printing The Sorted Array:-

1] a

2] b

3] c

4] k

5] n

6] v

4]Quick Sort Int:-

Input:- Pending……

Output:- Pending…..

Input:-

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

void Selection(int a[],int n);

void Input(int a[],int n);

void Print(int a[],int n);

int main() {

int n;

printf("Hello World!!!\nEnter The Size Of The Array:- ");

scanf("%d",&n);

int head[n];

Input(head,n);

Selection(head,n);

Print(head,n);

return 0;

}

void Selection(int a[],int n){

printf("Sorting!!!\n");

int i,j;

int temp;

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

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

if(a[i]>a[j]){

temp=a[i];

a[i]=a[j];

a[j]=temp;

}

}

}

}

void Input(int a[],int n){

printf("Enter The Values:- \n");

int i;

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

printf("%d:- ",i);

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

}

}

void Print(int a[],int n){

int i;

printf("Printing:- \n");

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

printf("%d:- %d\n",i,a[i]);

}

}

Output:-

Hello World!!!

Enter The Size Of The Array:- 10

Enter The Values:-

0:- 31

1:- 34

2:- 56

3:- 78

4:- 3

5:- 7

6:- 0

7:- 1

8:- 2

8:- 2

9:- 4

Sorting!!!

Sorting!!!

Printing:-

0:- 0

1:- 1

2:- 2

3:- 3

3:- 3

4:- 4

5:- 7

6:- 31

7:- 34

8:- 56

9:- 78

Input:-

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

void Selection(char \*a[],int n);

void Input(char \*a[],int n);

void Print(char \*a[],int n);

int main() {

int n;

printf("Hello World!!!\nEnter The Size Of The Array:- ");

scanf("%d",&n);

char \*head[n];

Input(head,n);

Selection(head,n);

Print(head,n);

return 0;

}

void Selection(char \*a[],int n){

printf("Sorting!!!\n");

int i,j;

char temp[10];

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

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

if(strcmp(a[i],a[j])>0){

strcpy(temp,a[i]);

strcpy(a[i],a[j]);

strcpy(a[j],temp);

}

}

}

}

void Input(char \*a[],int n){

printf("Enter The Values:- \n");

int i;

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

printf("%d:- ",i);

a[i]=(char \*)malloc(sizeof(char)\*8);

scanf("%s",a[i]);

}

}

void Print(char \*a[],int n){

int i;

printf("Printing:- \n");

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

printf("%d:- %s\n",i,a[i]);

}

}

Output:-

Hello World!!!

Enter The Size Of The Array:- 10

Enter The Values:-

0:- LL

1:- XX

2:- YY

3:- AA

4:- CC

5:- ZZ

6:- UU

7:- OO

8:- SS

9:- QQ

Sorting!!!

Printing:-

0:- AA

1:- CC

2:- LL

3:- OO

4:- QQ

5:- SS

6:- UU

7:- XX

8:- YY

9:- ZZ

Input:-

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

void input(int a[],int n);

int Linear(int a[],int n,int key);

int main() {

int n,key,position;

printf("Enter The Size Of The Array:- ");

scanf("%d",&n);

int head[n];

input(head,n);

printf("\nEnter The Key To Be Searched:- ");

scanf("%d",&key);

position=Linear(head,n,key);

if(position != -1){

printf("\nThe Key %d Is Found At %d position!!!",key,position);

}

else{

printf("\nThe Key %d Not Found In The Array!!!",key);

}

}

int Linear(int a[],int n,int key){

int i;

printf("Searching!!!\n");

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

if(a[i]==key){

return i;

}

}

return -1;

}

void input(int a[],int n){

int i;

printf("Enter The Values:- \n");

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

printf("%d:- ",i);

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

}

}

Output:-

Enter The Size Of The Array:- 10

Enter The Values:-

0:- 23

1:- 435

2:- 67

3:- 4

4:- 7

5:- 89

6:- 56

7:- 78

8:- 56

9:- 98

Enter The Key To Be Searched:- 67

Searching!!!

The Key 67 Is Found At 2 position!!!

Input:-

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

void input(char \*a[],int n);

int Linear(char \*a[],int n,char key[10]);

int main() {

int n,position;

char key[10];

printf("Enter The Size Of The Array:- ");

scanf("%d",&n);

char \*head[n];

input(head,n);

printf("\nEnter The Key To Be Searched:- ");

scanf("%s",key);

position=Linear(head,n,key);

if(position != -1){

printf("\nThe Key %s Is Found At %d position!!!",key,position);

}

else{

printf("\nThe Key %s Not Found In The Array!!!",key);

}

}

int Linear(char \*a[],int n,char key[10]){

int i;

printf("Searching!!!\n");

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

if(strcmp(a[i],key)==0){

return i;

}

}

return -1;

}

void input(char \*a[],int n){

int i;

printf("Enter The Values:- \n");

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

printf("%d:- ",i);

a[i]=(char \*)malloc(sizeof(char)\*8);

scanf("%s",a[i]);

}

}

Output:-

Enter The Size Of The Array:- 5

Enter The Values:-

0:- XX

1:- VV

2:- ZZ

3:- AA

4:- CC

Enter The Key To Be Searched:- AA

Searching!!!

The Key AA Is Found At 3 position!!!

ii]

Enter The Size Of The Array:- 4

Enter The Values:-

0:- AA

1:- KK

2:- LL

3:- SS

Enter The Key To Be Searched:- CC

Searching!!!

The Key CC Not Found In The Array!!!

Input:-

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

void Input(int a[],int n);

int Binary(int a[],int n,int key);

int main() {

int n,position;

int key;

printf("Enter The Size Of The Array:- ");

scanf("%d",&n);

int head[n];

Input(head,n);

printf("\nEnter The Key To Be Searched:- ");

scanf("%d",&key);

position=Binary(head,n,key);

if(position != -1){

printf("\nThe Key %d Is Found At %d position!!!",key,position);

}

else{

printf("\nThe Key %d Not Found In The Array!!!",key);

}

}

int Binary(int a[],int n,int key){

printf("Searching!!!\n");

int lb,ub,mid;

lb=0;

ub=n-1;

while(lb<=ub){

mid=(lb+ub)/2;

if(a[mid]==key){

return mid;

}

else if(a[mid]>key){

ub=mid-1;

}

else{

lb=mid+1;

}

}

return -1;

}

void Input(int a[],int n){

int i;

printf("Enter The Values:- \n");

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

printf("%d:- ",i);

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

}

}

Output:-

Enter The Size Of The Array:- 5

Enter The Values:-

0:- 1

1:- 2

2:- 3

3:- 4

4:- 5

Enter The Key To Be Searched:- 3

Searching!!!

The Key 3 Is Found At 2 position!!!

Input:-

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

void Input(char \*a[],int n);

int Binary(char \*a[],int n,char key[10]);

int main() {

int n,position;

char key[10];

printf("Enter The Size Of The Array:- ");

scanf("%d",&n);

char \*head[n];

Input(head,n);

printf("\nEnter The Key To Be Searched:- ");

scanf("%s",key);

position=Binary(head,n,key);

if(position != -1){

printf("\nThe Key %s Is Found At %d position!!!",key,position);

}

else{

printf("\nThe Key %s Not Found In The Array!!!",key);

}

}

int Binary(char \*a[],int n,char key[10]){

printf("Searching!!!\n");

int lb,ub,mid;

lb=0;

ub=n-1;

while(lb<=ub){

mid=(lb+ub)/2;

if(strcmp(a[mid],key)==0){

return mid;

}

else if(strcmp(a[mid],key)>0){

ub=mid-1;

}

else{

lb=mid+1;

}

}

return -1;

}

void Input(char \*a[],int n){

int i;

printf("Enter The Values:- \n");

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

printf("%d:- ",i);

a[i]=(char \*)malloc(sizeof(char)\*8);

scanf("%s",a[i]);

}

}

Output:-

Enter The Size Of The Array:- 5

Enter The Values:-

0:- AA

1:- BB

2:- CC

3:- DD

4:- EE

Enter The Key To Be Searched:- EE

Searching!!!

The Key EE Is Found At 4 position!!!

ii]

Enter The Size Of The Array:- 5

Enter The Values:-

0:- AA

1:- BB

2:- CC

3:- DD

4:- EE

Enter The Key To Be Searched:- FF

Searching!!!

The Key FF Not Found In The Array!!!

Input:-

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

void Input(int a[],int n);

int BinaryR(int a[],int n,int key,int lb,int ub);

int main() {

int n,position;

int key;

printf("Enter The Size Of The Array:- ");

scanf("%d",&n);

int head[n];

Input(head,n);

printf("\nEnter The Key To Be Searched:- ");

scanf("%d",&key);

position=BinaryR(head,n,key,0,n-1);

if(position != -1){

printf("\nThe Key %d Is Found At %d position!!!",key,position);

}

else{

printf("\nThe Key %d Not Found In The Array!!!",key);

}

}

int BinaryR(int a[],int n,int key,int lb,int ub){

int mid;

while(lb<=ub){

mid=(lb+ub)/2;

if(a[mid]==key){

return mid;

}

else if(a[mid]>key){

return BinaryR(a,n,key,lb,mid-1);

}

else{

return BinaryR(a,n,key,mid+1,ub);

}

}

return -1;

}

void Input(int a[],int n){

int i;

printf("Enter The Values:- \n");

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

printf("%d:- ",i);

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

}

}

Output:-

Enter The Size Of The Array:- 5

Enter The Values:-

0:- 1

1:- 2

2:- 3

3:- 4

4:- 5

Enter The Key To Be Searched:- 3

The Key 3 Is Found At 2 position!!!

Input:-

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

void Input(char \*a[],int n);

int BinaryR(char \*a[],int n,char key[10],int lb,int ub);

int main() {

int n,position;

char key[10];

printf("Enter The Size Of The Array:- ");

scanf("%d",&n);

char \*head[n];

Input(head,n);

printf("\nEnter The Key To Be Searched:- ");

scanf("%s",key);

position=BinaryR(head,n,key,0,n-1);

if(position != -1){

printf("\nThe Key %s Is Found At %d position!!!",key,position);

}

else{

printf("\nThe Key %s Not Found In The Array!!!",key);

}

}

int BinaryR(char \*a[],int n,char key[10],int lb,int ub){

int mid;

while(lb<=ub){

mid=(lb+ub)/2;

if(strcmp(a[mid],key)==0){

return mid;

}

else if((strcmp(a[mid],key))>0){

return BinaryR(a,n,key,lb,mid-1);

}

else{

return BinaryR(a,n,key,mid+1,ub);

}

}

return -1;

}

void Input(char \*a[],int n){

int i;

printf("Enter The Values:- \n");

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

printf("%d:- ",i);

a[i]=(char \*)malloc(sizeof(char)\*8);

scanf("%s",a[i]);

}

}

Output:-

Enter The Size Of The Array:- 5

Enter The Values:-

0:- AA

1:- BB

2:- CC

3:- DD

4:- EE

Enter The Key To Be Searched:- DD

The Key DD Is Found At 3 position!!!

Input:-

#include<stdio.h>

#include<stdlib.h>

#define Newnode (struct node \*)malloc(sizeof(struct node));

struct node {

int coeff;

int power;

struct node \*next;

};

struct node \* Create(int n){

struct node \*t=NULL,\*f=NULL,\*l=NULL;

int i;

if(n<=0){

printf("You Have Entered Less Than Or Equal To 0 For Number Of Polynomials!!!\n");

}

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

t=Newnode;

t->next=NULL;

printf("Enter The Coefficient:- ");

scanf("%d",&t->coeff);

printf("Enter The Power:- ");

scanf("%d",&t->power);

if(f==NULL){

f=t;

l=t;

}

else{

l->next=t;

l=l->next;

}

}

return f;

}

void Print(struct node \*poly){

struct node \*t=NULL;

for(t=poly;t!=NULL;t=t->next){

printf("%dx^%d",t->coeff,t->power);

if(t->next != NULL){

printf("+");

}

}

printf("\n");

}

struct node \* AdditionPoly(struct node \*poly1,struct node \*poly2){

struct node \*t=NULL,\*fpoly=NULL,\*spoly=NULL,\*hpoly=NULL,\*hlast=NULL;

fpoly=poly1;

spoly=poly2;

while(fpoly != NULL && spoly != NULL){

t=Newnode;

t->next=NULL;

if(fpoly->power==spoly->power){

t->coeff=fpoly->coeff+spoly->coeff;

t->power=fpoly->power;

fpoly=fpoly->next;

spoly=spoly->next;

}

else if (fpoly->power>spoly->power)

{

t->coeff=fpoly->coeff;

t->power=fpoly->power;

fpoly=fpoly->next;

}

else {

t->coeff=spoly->coeff;

t->power=spoly->power;

spoly=spoly->next;

}

if(hpoly==NULL){

hpoly=t;

hlast=t;

}

else{

hlast->next=t;

hlast=hlast->next;

}

}

while(fpoly!= NULL){

t=Newnode;

t->next=NULL;

t->coeff=fpoly->coeff;

t->power=fpoly->power;

fpoly=fpoly->next;

if(hpoly==NULL){

hpoly=t;

hlast=t;

}

else {

hlast->next=t;

hlast=hlast->next;

}

}

while(spoly!= NULL){

t=Newnode;

t->next=NULL;

t->coeff=spoly->coeff;

t->power=spoly->power;

spoly=spoly->next;

if(hpoly==NULL){

hpoly=t;

hlast=t;

}

else {

hlast->next=t;

hlast=hlast->next;

}

}

return hpoly;

}

int main(){

int n1,n2;

struct node \*poly1=NULL,\*poly2=NULL,\*headp=NULL;

printf("Enter The Size Of The first Polynomial:- ");

scanf("%d",&n1);

poly1=Create(n1);

printf("Enter The Size Of The Second Polynomial:- ");

scanf("%d",&n2);

poly2=Create(n2);

printf("\t\t\t");

Print(poly1);

printf("\t\t+\t");

Print(poly2);

printf("\t -------------------------------------------------------------\t\n");

headp=AdditionPoly(poly1,poly2);

printf("\t\t\t");

Print(headp);

printf("\t\t\t\t\t\t\t\t\t\t By:- Cdt. Akash Kalidas Durane");

}

Output:-

i]

Enter The Size Of The first Polynomial:- 4

Enter The Coefficient:- 4

Enter The Power:- 3

Enter The Coefficient:- 6

Enter The Power:- 2

Enter The Coefficient:- 7

Enter The Power:- 1

Enter The Coefficient:- 9

Enter The Power:- 0

Enter The Size Of The Second Polynomial:- 4

Enter The Coefficient:- 4

Enter The Power:- 3

Enter The Coefficient:- 6

Enter The Power:- 2

Enter The Coefficient:- 8

Enter The Power:- 1

Enter The Coefficient:- 8

Enter The Power:- 0

4x^3+6x^2+7x^1+9x^0

+ 4x^3+6x^2+8x^1+8x^0

-------------------------------------------------------------

8x^3+12x^2+15x^1+17x^0

By:- Cdt. Akash Kalidas Durane

ii]

Enter The Size Of The first Polynomial:- 4

Enter The Coefficient:- 3

Enter The Power:- 53

Enter The Coefficient:- 6

Enter The Power:- 3

Enter The Coefficient:- 6

Enter The Power:- 4

Enter The Coefficient:- 5

Enter The Power:- 3

Enter The Size Of The Second Polynomial:- 0

You Have Entered Less Than Or Equal To 0 For Number Of Polynomials!!!

3x^53+6x^3+6x^4+5x^3

+

-------------------------------------------------------------

3x^53+6x^3+6x^4+5x^3

By:- Cdt. Akash Kalidas Durane

Input:-

#include<stdio.h>

#include<stdlib.h>

#define Newnode (struct node \*)malloc(sizeof(struct node))

#define Start NULL

struct node \*root;

int cnt;

struct node{

struct node \*LC;

int data;

struct node \*RC;

};

void init(){

root=Start;

}

void BST(int item){

struct node \*t=NULL,\*t1=NULL,\*t2=NULL;

t=Newnode;

t->data=item;

t->LC=Start;

t->RC=Start;

if(root==Start){

root=t;

}

else{

t1=root;

while(t1 != NULL){

t2=t1;

if(item<=t1->data){

t1=t1->LC;

}

else{

t1=t1->RC;

}

}

if(item<=t2->data){

t2->LC=t;

}

else{

t2->RC=t;

}

}

}

int BinarySearch(struct node \*t,int key){

while(t != NULL){

if(t->data==key){

return 1;

}

else if(key<t->data){

t=t->LC;

}

else{

t=t->RC;

}

}

return 0;

}

void Preorder(struct node \*t){

if(t != NULL){

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

Preorder(t->LC);

Preorder(t->RC);

}

}

void Inorder(struct node \*t){

if(t != NULL){

Inorder(t->LC);

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

Inorder(t->RC);

}

}

void Postorder(struct node \*t){

if(t != NULL){

Postorder(t->LC);

Postorder(t->RC);

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

}

}

int CountLeaf(struct node \*t){

if(t!=NULL){

CountLeaf(t->LC);

CountLeaf(t->RC);

if(t->LC==NULL && t->RC==NULL){

cnt++;

}

}

}

int CountNonLeaf(struct node \*t){

if(t!=NULL){

CountNonLeaf(t->LC);

CountNonLeaf(t->RC);

if(t->LC!=NULL || t->RC!=NULL){

cnt++;

}

}

}

int CountNode(struct node \*t){

if(t!=NULL){

CountNode(t->LC);

CountNode(t->RC);

cnt++;

}

}

int main() {

int item,i,n,key;

printf("Enter The Number Of Values You're Going To Enter:- ");

scanf("%d",&n);

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

printf("Enter The Item:- ");

scanf("%d",&item);

BST(item);

}

printf("Enter The Key To Be Serched:- ");

scanf("%d",&key);

if(BinarySearch(root,key)==1){

printf("The Key %d Is Present In The Tree\n",key);

}

else{

printf("Key %d Not Found In The Tree\n",key);

}

printf("The Pre-Order Of The Tree Are:- ");

Preorder(root);

printf("\nThe In-Order Of The Tree Are:- ");

Inorder(root);

printf("\nThe Post-Order Of The Tree Are:- ");

Postorder(root);

cnt=0;

CountLeaf(root);

printf("\nThe Number Of Leaf Node's Are:- %d\n",cnt);

cnt=0;

CountNonLeaf(root);

printf("The Number Of Non-Leaf Node's Are:- %d\n",cnt);

cnt=0;

CountNode(root);

printf("The Total Number Of Nodes Are:- %d\n",cnt);

}

Output:-

Enter The Number Of Values You're Going To Enter:- 8

Enter The Item:- 2

Enter The Item:- 17

Enter The Item:- 47

Enter The Item:- 5

Enter The Item:- 1

Enter The Item:- 8

Enter The Item:- 99

Enter The Item:- 0

Enter The Key To Be Serched:- 1

The Key 1 Is Present In The Tree

The Pre-Order Of The Tree Are:- 2 1 0 17 5 8 47 99

The In-Order Of The Tree Are:- 0 1 2 5 8 17 47 99

The Post-Order Of The Tree Are:- 0 1 8 5 99 47 17 2

The Number Of Leaf Node's Are:- 3

The Number Of Non-Leaf Node's Are:- 5

The Total Number Of Nodes Are:- 8

Input:-

#include<stdio.h>

#include<stdlib.h>

#define Size 5

#define Start -1

int Q[Size],Front,Rear;

void init(){

Front=Start;

Rear=Start;

}

int isEmpty(){

if(Front==Rear){

return 1;

}

else{

return 0;

}

}

int isFull(){

if(Rear==Size-1){

return 1;

}

else{

return 0;

}

}

void AddQ(int data){

if(!isFull()){

Rear++;

Q[Rear]=data;

printf("%d Added Succesfully In The Queue!!!\n",data);

}

}

int DelQ(){

if(!isEmpty()){

Front++;

return Q[Front];

}

}

int PeekQ(){

if(!isEmpty()){

return Q[Front+1];

}

}

int main(){

int choice,data;

while(1){

printf("Menu:- \n");

printf("1. ADD\n");

printf("2. DELETE\n");

printf("3. PEEK\n");

printf("4. EXIT\n");

printf("Enter You're Choice:- ");

scanf("%d",&choice);

switch (choice)

{

case 1:

printf("\nEnter The Data You Wan't To Add:- ");

scanf("%d",&data);

if(isFull()){

printf("Queue Is Full!!!\n");

}

else{

AddQ(data);

}

break;

case 2:

if(isEmpty()){

printf("Queue Is Empty!!!\n");

}

else{

data=DelQ();

printf("The Data %d Is Deleted From The Queue\n",data);

}

break;

case 3:

if(isEmpty()){

printf("Queue Is Empty!!!\n");

}

else{

data=PeekQ();

printf("The Peek Element Is %d!!!\n",data);

}

break;

case 4:

exit(0);

default:

printf("Invalid Chice!!!!\n");

break;

}

}

return 0;

}

Output:-

Menu:-

1. ADD

2. DELETE

3. PEEK

4. EXIT

Enter You're Choice:- 1

Enter The Data You Wan't To Add:- 11

11 Added Succesfully In The Queue!!!

Menu:-

1. ADD

2. DELETE

3. PEEK

4. EXIT

Enter You're Choice:- 1

Enter The Data You Wan't To Add:- 22

22 Added Succesfully In The Queue!!!

Menu:-

1. ADD

2. DELETE

3. PEEK

4. EXIT

Enter You're Choice:- 1

Enter The Data You Wan't To Add:- 33

33 Added Succesfully In The Queue!!!

Menu:-

1. ADD

2. DELETE

3. PEEK

4. EXIT

Enter You're Choice:- 2

The Data 11 Is Deleted From The Queue

Menu:-

1. ADD

2. DELETE

3. PEEK

4. EXIT

Enter You're Choice:- 22

Invalid Chice!!!!

Menu:-

1. ADD

2. DELETE

3. PEEK

4. EXIT

Enter You're Choice:- 3

The Peek Element Is 22!!!

Menu:-

1. ADD

2. DELETE

3. PEEK

4. EXIT

Enter You're Choice:- 2

The Data 22 Is Deleted From The Queue

Menu:-

1. ADD

2. DELETE

3. PEEK

4. EXIT

Enter You're Choice:- 3

The Peek Element Is 33!!!

Menu:-

1. ADD

2. DELETE

3. PEEK

4. EXIT

Enter You're Choice:- 2

The Data 33 Is Deleted From The Queue

Menu:-

1. ADD

2. DELETE

3. PEEK

4. EXIT

Enter You're Choice:- 2

Queue Is Empty!!!

Menu:-

1. ADD

2. DELETE

3. PEEK

4. EXIT

Enter You're Choice:- 4

Input:-

#include<stdio.h>

#include<stdlib.h>

#define Newnode (struct node \*)malloc(sizeof(struct node))

#define Start NULL

struct node {

int data;

struct node \*next;

};

struct node \*Front=NULL,\*Rear=NULL;

void init(){

Front=Start;

Rear=Start;

}

int isEmpty(){

if(Front==NULL){

return 1;

}

else{

return 0;

}

}

void AddQ(int data){

struct node \*t;

t=Newnode;

if(t==NULL){

printf("Queue Is over Flow!!!" );

}

else{

t->data=data;

t->next=NULL;

if(Rear==NULL){

Rear=t;

Front=t;

}

else{

Rear->next=t;

Rear=Rear->next;

}

}

printf("The Data %d Added Successfully!!!",data);

}

int DelQ(){

int num;

struct node \*t;

if(!isEmpty()){

num=Front->data;

if(Front->next==NULL){

free(Front);

Front=Rear=NULL;

return num;

}

else{

t=Front;

Front=Front->next;

free(t);

return num;

}

}

}

int PeekQ(){

struct node \*t;

int num;

if(!isEmpty()){

num=Front->data;

return num;

}

}

int main(){

int choice,data;

init();

while(1){

printf("Menu:- \n");

printf("1. ADD\n");

printf("2. DELETE\n");

printf("3. PEEK\n");

printf("4. EXIT\n");

printf("Enter You're Choice:- ");

scanf("%d",&choice);

switch (choice)

{

case 1:

printf("\nEnter The Data You Wan't To Add:- ");

scanf("%d",&data);

AddQ(data);

break;

case 2:

if(isEmpty()){

printf("Queue Is Empty!!!\n");

}

else{

data=DelQ();

printf("The Data %d Is Deleted From The Queue\n",data);

}

break;

case 3:

if(isEmpty()){

printf("Queue Is Empty!!!\n");

}

else{

data=PeekQ();

printf("The Peek Element Is %d!!!\n",data);

}

break;

case 4:

exit(0);

default:

printf("Invalid Chice!!!!\n");

break;

}

}

return 0;

}

output:-

Menu:-

1. ADD

2. DELETE

3. PEEK

4. EXIT

Enter You're Choice:- 1

Enter The Data You Wan't To Add:- 11

The Data 11 Added Successfully!!!Menu:-

1. ADD

2. DELETE

3. PEEK

4. EXIT

Enter You're Choice:- 1

Enter The Data You Wan't To Add:- 22

The Data 22 Added Successfully!!!Menu:-

1. ADD

2. DELETE

3. PEEK

4. EXIT

Enter You're Choice:- 1

Enter The Data You Wan't To Add:- 33

The Data 33 Added Successfully!!!Menu:-

1. ADD

2. DELETE

3. PEEK

4. EXIT

Enter You're Choice:- 2

The Data 11 Is Deleted From The Queue

Menu:-

1. ADD

2. DELETE

3. PEEK

4. EXIT

Enter You're Choice:- 3

The Peek Element Is 22!!!

Menu:-

1. ADD

2. DELETE

3. PEEK

4. EXIT

Enter You're Choice:- 2

The Data 22 Is Deleted From The Queue

Menu:-

1. ADD

2. DELETE

3. PEEK

4. EXIT

Enter You're Choice:- 2

The Data 33 Is Deleted From The Queue

Menu:-

1. ADD

2. DELETE

3. PEEK

4. EXIT

Enter You're Choice:- 1

Enter The Data You Wan't To Add:- 11

The Data 11 Added Successfully!!!Menu:-

1. ADD

2. DELETE

3. PEEK

4. EXIT

Enter You're Choice:- 22

Invalid Chice!!!!

Menu:-

1. ADD

2. DELETE

3. PEEK

4. EXIT

Enter You're Choice:- 2

The Data 11 Is Deleted From The Queue

Menu:-

1. ADD

2. DELETE

3. PEEK

4. EXIT

Enter You're Choice:- 3

Queue Is Empty!!!

Menu:-

1. ADD

2. DELETE

3. PEEK

4. EXIT

Enter You're Choice:- 4

Input:-

#include<stdio.h>

#include<stdlib.h>

#define Newnode (struct node \*)malloc(sizeof(struct node))

#define Size 30

#define Bottom -1

#define Operator 5

struct Ptab{

char opr;

int priority;

};

struct Ptab OpTab[Operator]= {

{'$',3},

{'\*',2},

{'/',2},

{'+',1},

{'-',1}

};

char Stack[Size];

int Top;

void init(){

Top=Bottom;

}

int isEmpty(){

if(Top==Bottom){

return 1;

}

else{

return 0;

}

}

int isFull(){

if(Top==Size-1){

return 1;

}

else{

return 0;

}

}

void Push(char data){

if(isFull()){

printf("Stack Is Full!!!");

}

else{

Top++;

Stack[Top]=data;

}

}

int pop(){

char Data;

if(isEmpty()){

printf("Stack Is Empty!!!");

}

else{

Data=Stack[Top];

Top--;

return Data;

}

}

int peek(){

if(isEmpty()){

printf("Stack Is Empty!!!");

}

else{

return Stack[Top];

}

}

int GetPriority(char Opr){

for(int i=0;i<Operator;i++){

if(OpTab[i].opr==Opr){

return OpTab[i].priority;

}

}

return -1;

}

int main(){

char Exp[50];

printf("Enter The Infix Expression:- ");

scanf("%s",Exp);

init();

printf("The Postfix Expression Is:- ");

for(int i=0;Exp[i]!='\0';i++){

if(GetPriority(Exp[i])==-1){

printf("%c",Exp[i]);

}

else{

while(!isEmpty()){

if(GetPriority(peek())>=GetPriority(Exp[i])){

printf("%c",pop());

}

else{

break;

}

}

Push(Exp[i]);

}

}

while(!isEmpty()){

printf("%c",pop());

}

}

Output:-

Enter The Infix Expression:- a+e-f\*c/d

The Postfix Expression Is:- ae+fc\*d/-