

Data Structure

Lab File

|  |  |
| --- | --- |
| Submitted to | Submitted by |
| Dr. Anu Taneja | Prachi Sharma  MCA 2nd semester |

Index

|  |  |  |
| --- | --- | --- |
| Sr. no | Content | Page no |
| 1 | Insertion Operation |  |
| 2 | Deletion Operation |  |
| 3 i | Linear search with first occurence |  |
| ii | Linear search with multiple occurence |  |
| 4 i | Binary search with recursion |  |
| ii | Binary search without recursion |  |
| 5 | Find maximum element |  |
| 6 | Find second maximum element |  |
| 7 | Bubble sort |  |
| 8 | Selection sort |  |
| 9 | Insertion sort |  |
| 10 | Counting sort |  |
| 11 | Radix Sort |  |
| 12 | Menu driven program   * 1. To add two matrices   2. To subtract two matrices   3. To multiply two matrices |  |
| 13 | Identity matrix |  |
| 14 | Identical matrix |  |
| 15 | Print diagonal elements |  |
| 16 | Sum of all elements |  |
| 17 | Row triplet |  |
| 18 | Check Sparse matrix |  |
| 19 | Arrange all rows in ascending order |  |
| 20 | Missing element |  |
| 21 | Duplicate elements |  |
| 22 | Remove duplicate elements |  |
| 23 | Find sum using free  i.Calloc  ii.Malloc  iii.Realloc |  |
| 24 | Linked list Insertion  At beginning  At end  iii At specific position |  |
| 25 | Linked list deletion  At beginning  At end  iii At specific position |  |
| 26 | Doubly Linked list insertion  At beginning  At end  iii at specific position |  |
| 27 |  |  |
| 28 |  |  |
| 29 |  |  |
| 30 |  |  |
| 31 |  |  |
| 32 |  |  |
| 33 |  |  |
| 34 |  |  |
| 35 |  |  |
| 36 |  |  |
| 37 |  |  |
| 38 |  |  |
| 39 |  |  |
| 40 |  |  |
| 41 |  |  |
| 42 |  |  |
| 43 |  |  |
| 44 |  |  |
| 45 |  |  |
| 46 |  |  |
| 47 |  |  |

1. **WAP to insert an element in array.**

**Source code**:

#include<stdio.h>

void main()

{

int n ,a[10],c,i,j,pos;

printf("enter the total no of element : ");

scanf("%d" ,&n);

printf("enter array elements : ");

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

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

printf("enter the element to be inserted: ");

scanf("%d" ,&c);

printf("enter the position to which element has to be inserted : ");

scanf("%d" ,&pos);

for(i=n-1;i>=pos;i--)

{

a[i+1]=a[i];

}

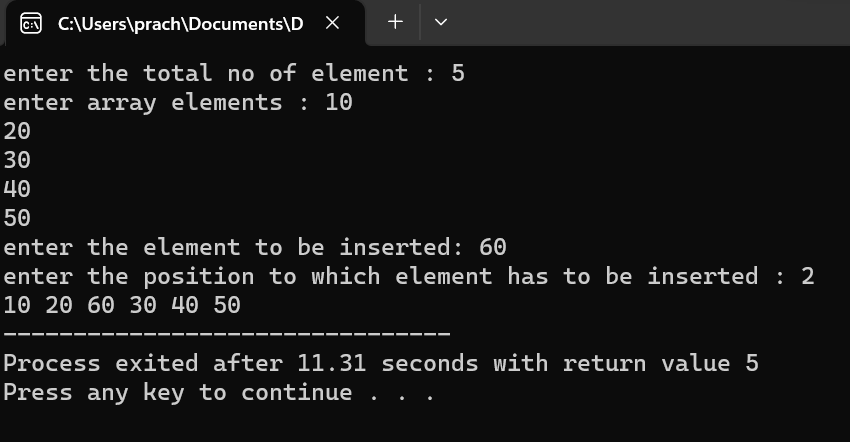
a[pos]=c;

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

printf("%d ",a[i]);

}

**Output :**



1. **WAP to perform Deletion operation**.

**Source code :**

#include<stdio.h>

void main()

{

int n ,a[10],c,i,j,pos;

printf("enter the total no of element : ");

scanf("%d" ,&n);

printf("enter array elements : ");

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

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

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

{

a[i]=a[i+1];

}

n--;

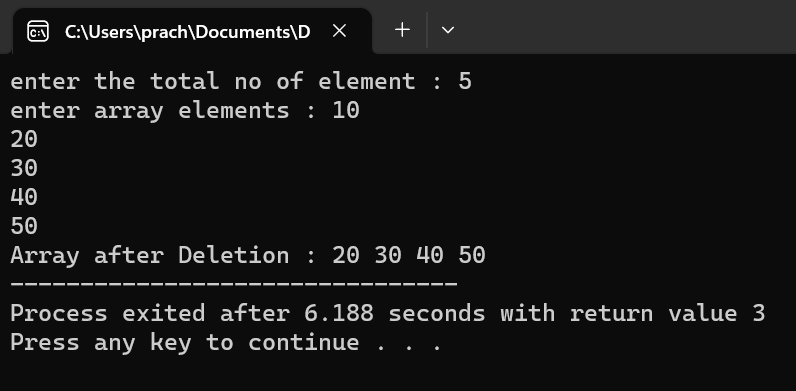
printf("Array after Deletion : ");

for(i=0;i<4;i++)

printf("%d ",a[i]);

}

**Output :**



1. **i. WAP to perform Linear Search operation with single occurrence .**

**Source code :**

void main()

{

int n ,a[6],c,i,flag;

printf("enter the total no of element : ");

scanf("%d" ,&n);

printf("enter array elements : ");

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

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

printf("enter the value to be searched : ");

scanf("%d",&c);

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

{

if(a[i]==c)

{

printf("Linear search with single occurence ");

printf("\n Number found at location : %d",i);

flag++;

break;

}

}

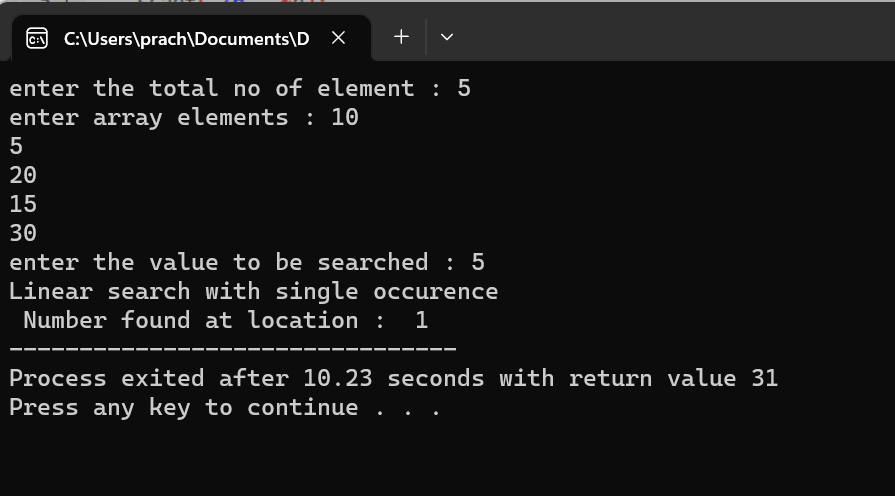
if(flag==0)

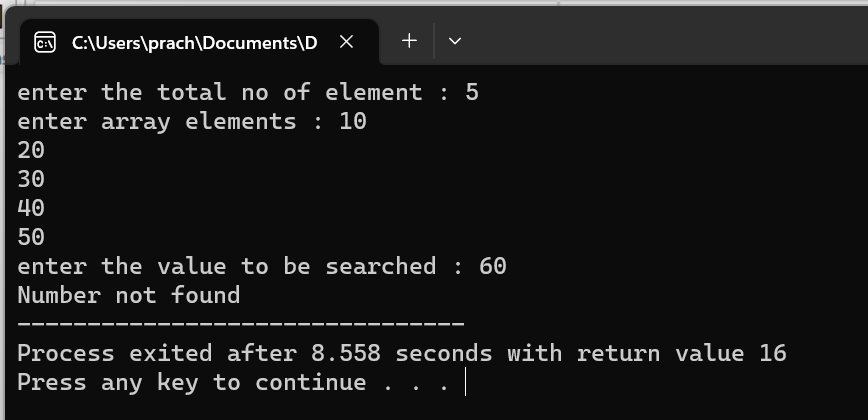
{

printf("Number not found");

} }

**Output :**

****

****

1. **WAP to perform Linear Search operation with multiple occurrence and also count the occurence .**

**Source code :**

void main()

{

int n ,a[10],c,i,flag=0;

printf("enter the total no of element : ");

scanf("%d" ,&n);

printf("enter array elements : ");

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

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

printf("enter the value to be searched : ");

scanf("%d",&c);

printf("Linear search with multiple occurence ");

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

{

if(a[i]==c)

{

printf("\n Number found at location : %d",i);

flag++;

}

}

printf("\n number occured at %d times ",flag);

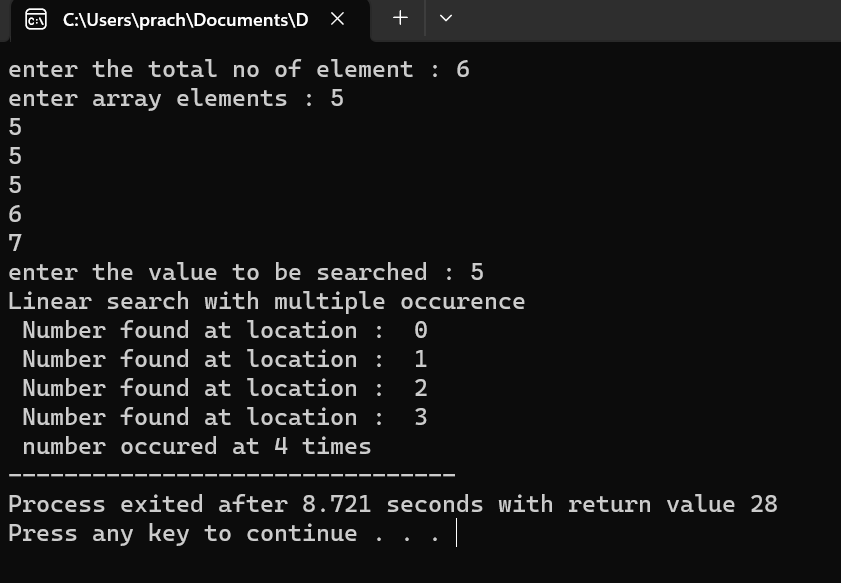
if(flag==0)

{

printf("Number not found");

}}

**Output :**

****

1. **i. WAP to perform Binary Search with Recursion.**

**Source code :**

#include<stdio.h>

int binarysearch(int a[],int lb,int ub,int c )

{

if(lb<=ub)

{

int mid=(lb+ub)/2;

if(a[mid]<c)

return binarysearch(a,mid+1,ub,c);

else if(a[mid]==c)

return mid;

else

return binarysearch(a,lb,mid-1,c);

}

else

return -1;

}

int main()

{

int arr[6],i,n,c;

printf("enter the total no of element : ");

scanf("%d" ,&n);

printf("enter array elements : ");

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

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

printf("enter the value to be searched : ");

scanf("%d",&c);

// int arr[]={5,10,15,20,25};

int result = binarysearch(arr, 0,n-1,c);

if(result == -1)

printf("Element is not present in array");

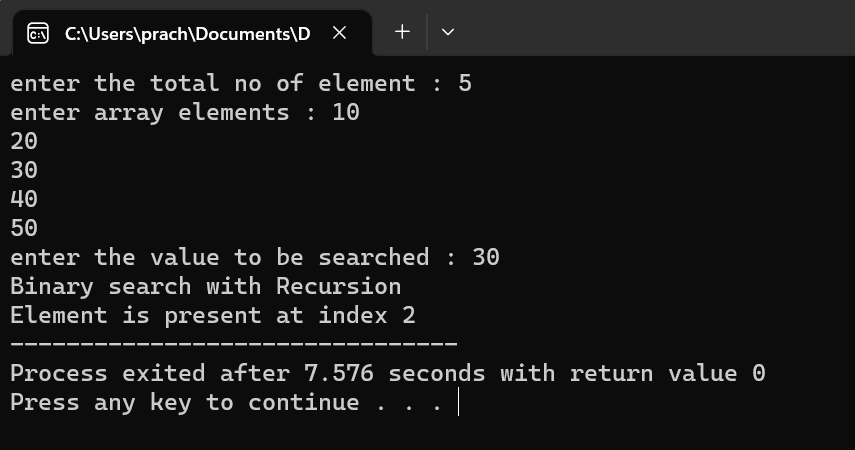
else

printf("Element is present at index %d", result);

return 0;

}

**Output :**

****

1. **WAP to perform Binary Search without Recursion.**

**Source code :**

#include<stdio.h>

void main()

{

int n ,a[6],c,i,mid,lb,ub;

printf("enter the total no of element : ");

scanf("%d" ,&n);

printf("enter array elements : ");

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

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

printf("enter the value to be searched : ");

scanf("%d",&c);

lb=0;

ub=n-1;

mid=(lb+ub)/2;

while(lb<=ub)

{

if(a[mid]<c)

lb=mid+1;

else if(a[mid]==c)

{

printf("%d no found at location %d",c,mid+1);

break;

}

else

ub=mid-1;

mid=(lb+ub)/2;

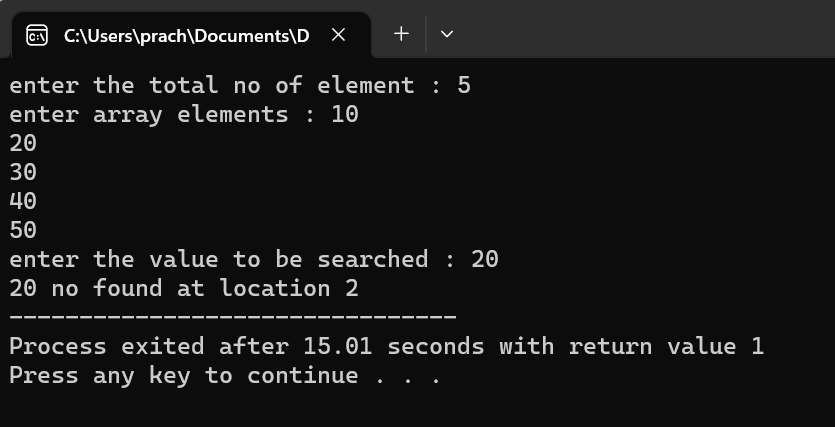
}

if(lb>ub)

printf("the no not found",i);

}

**Output :**

****

1. **WAP to find maximum element in the array.**

**Source code :**

#include<stdio.h>

void main()

{

int n ,a[10],i,max=a[0];

printf("enter the total no of element : ");

scanf("%d" ,&n);

printf("enter array elements : ");

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

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

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

{

if(a[i]>max)

{

max=a[i];

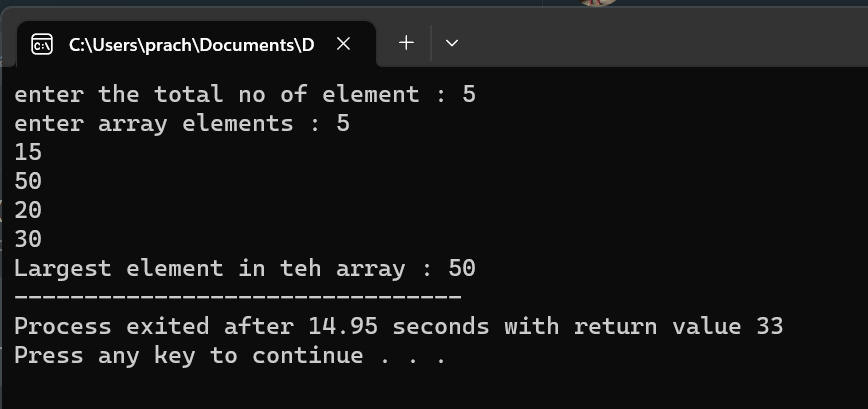
}

}

printf("Largest element in teh array : %d",max);

}

**Output :**

****

1. **WAP to find second maximum element in the array.**

**Source code :**

#include<stdio.h>

void main()

{

int n ,a[10],i,sl,max;

sl=max=0;

printf("enter the total no of element : ");

scanf("%d" ,&n);

printf("enter array elements : ");

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

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

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

{

if(a[i]>max)

{

sl=max;

max=a[i];

}

else if (a[i]<max && a[i]>sl)

{

sl=a[i];

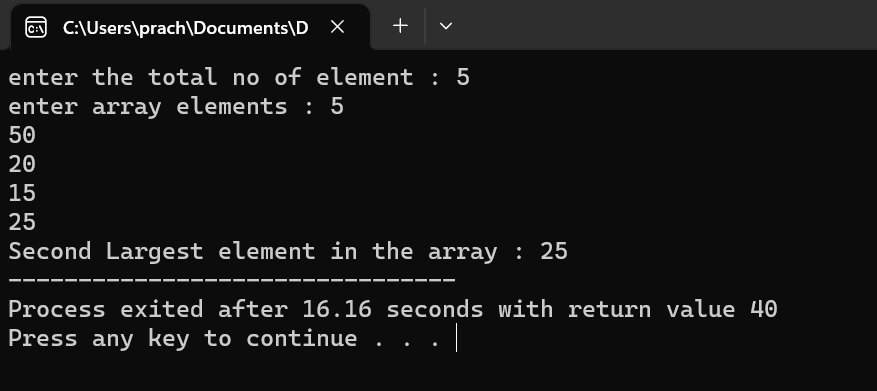
}

}

printf("Second Largest element in the array : %d",sl);

}

**Output :**

****

1. **WAP to perform Bubble Sort**

**Source code :**

#include<stdio.h>

void main()

{

int n ,a[6],c,i,j,temp;

printf("enter the total no of element : ");

scanf("%d" ,&n);

printf("enter array elements : ");

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

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

for(i=0;i<n-1;i++)

{

for(j=0;j<n-1;j++)

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

{

temp=a[j];

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

a[j+1]=temp;

}

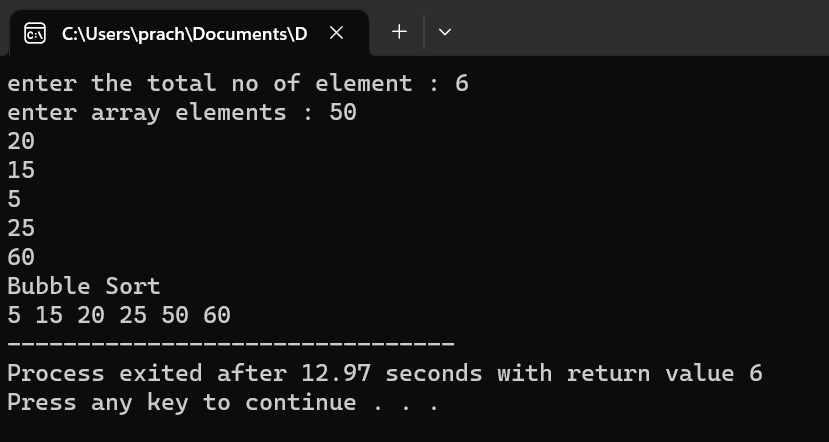
}

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

printf("%d ",a[i]);

}

**Output :**

****

1. **WAP to perform Selection Sort.**

**Source code :**

#include <stdio.h>

int main() {

int a[10];

int n=5,min,i,j,temp;

printf("enter the total no of element : ");

scanf("%d" ,&n);

printf("enter array elements : ");

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

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

for(i=0;i<n-1;i++)

{

min=i;

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

{

if(a[j]<a[min])

{

min=j;

}

}

if(min!=i)

{

temp=a[i];

a[i]=a[min];

a[min]=temp;

}

}

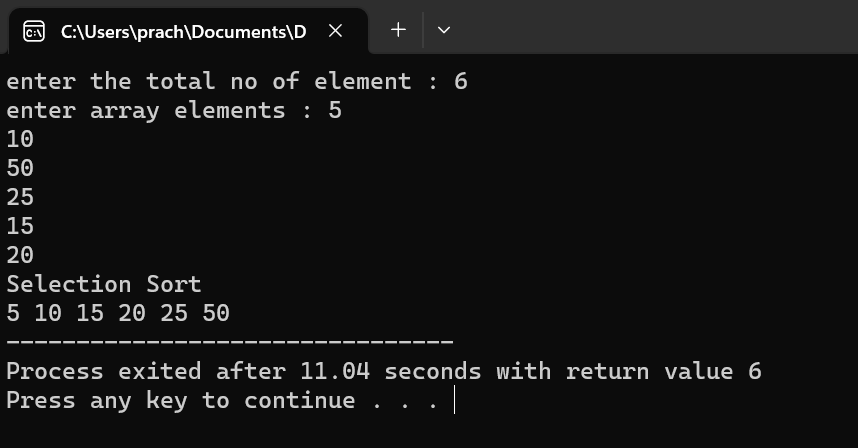
printf("Selection Sort \n");

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

printf("%d ",a[i]);

}

**Output :**

****

1. **WAP to perform Insertion Sort.**

**Source code :**

#include<stdio.h>

void main(){

int n ,a[6],c,i,j,temp;

printf("enter the total no of element : ");

scanf("%d" ,&n);

printf("enter array elements : ");

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

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

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

temp=a[i];

j=i-1;

while(j>=0 && a[j]>temp){

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

j=j-1;}

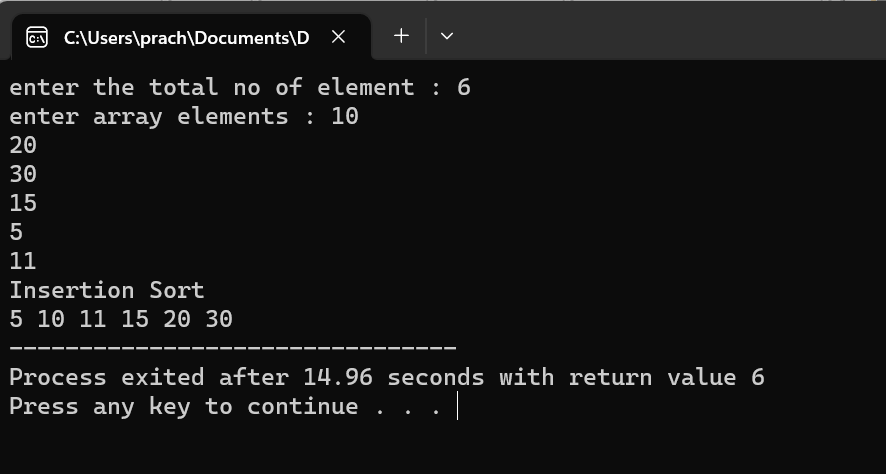
a[j+1]=temp;}

printf("Insertion Sort \n");

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

printf("%d ",a[i]);}

**Output :**

****

**10.WAP to perform Counting Sort.**

**Source code :**

#include<stdio.h>

void main()

{

int n,i,max;

printf("enter the total no of element : ");

scanf("%d" ,&n);

int a[n];

printf("enter array elements : ");

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

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

printf("The array is ");

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

printf("%d ", a[i]);

}

max=a[0];

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

{

if(a[i]>max)

{

max=a[i];

}

}

int count[max+1];

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

{

count[i] = 0;

}

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

{

count[a[i]]++;

}

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

{

count[i]=count[i]+count[i-1];

}

int output[n];

for(i=n-1;i>=0;i--)

{

output[count[a[i]]-1]=a[i];

count[a[i]]--;

}

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

a[i] = output[i];

}

printf("\nThe sorted array is ");

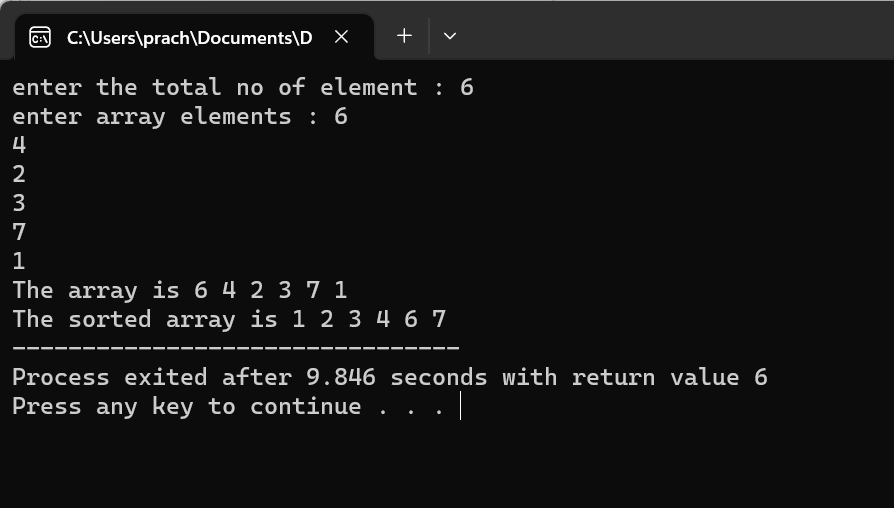
for (i = 0; i < n; i++)

{

printf("%d ", a[i]); }

}

**Output :**

****

1. **WAP to perform Radix Sort.**

**Source code :**

#include <stdio.h>

void counting\_sort(int a[],int n,int pos)

{

int count[10]={0};

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

{

count[(a[i]/pos)%10]++;

}

for(int i=1;i<10;i++)

{

count[i]=count[i]+count[i-1];

}

int o[n];

for(int i=n-1;i>=0;i--)

{

o[count[(a[i]/pos)%10]-1]=a[i];

count[(a[i]/pos)%10]--;

}

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

a[i]=o[i];

}

int main() {

int a[10],n,max;

printf("enter the no of elements : ");

scanf("%d",&n);

printf("enter the array elements : ");

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

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

max=a[0];

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

{

if(a[i]>max)

{

max=a[i];

}

}

int pos;

for(pos=1;max/pos>0;pos=pos\*10)

{

counting\_sort(a,n,pos);

}

printf("Sorted array: ");

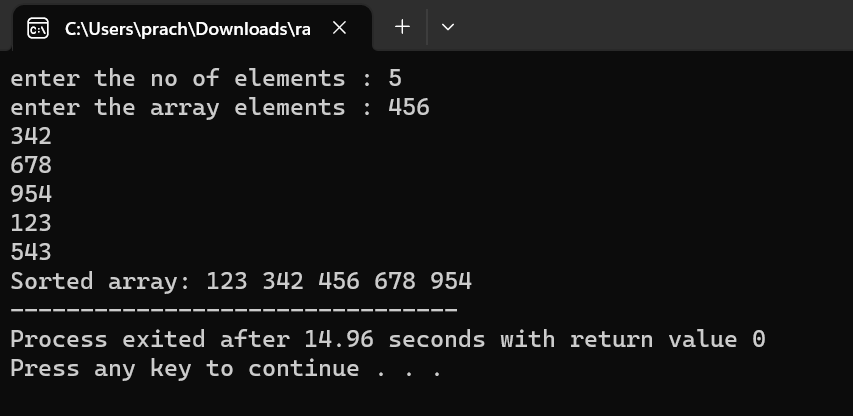
for(int i=0;i<n;i++){

printf("%d ",a[i]);

}

}

**Output:**

****

1. **Write a menu driven program to perform Addition,Subtraction,Multiplication on two dimensional array .**

**Source code :**

#include<stdio.h>

int main()

{

int i,j,d,k,c[2][2],m;

int a[2][2]={1,2,3,4};

int b[2][2]={5,6,7,8};

here:

printf("1 for sum\n");

printf("2 for subtract\n");

printf("3 for multiply\n");

printf("enter your choice : ");

scanf("%d",&d);

switch(d){

case 1:

for(i=0;i<2;i++)

{

for(j=0;j<2;j++)

{

c[i][j]=a[i][j]+b[i][j];

}}

printf("the sum is : \n ");

for(i=0;i<2;i++)

{

for(j=0;j<2;j++)

{

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

}

printf("\n");

}

break;

case 2:

for(i=0;i<2;i++)

{

for(j=0;j<2;j++)

{

c[i][j]=a[i][j]-b[i][j];

}

}

printf("the difference is : \n");

for(i=0;i<2;i++)

{

for(j=0;j<2;j++)

{

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

}

printf("\n");

}

break;

case 3:

for(i=0;i<2;i++)

{

for(j=0;j<2;j++)

{

c[i][j]=0;

for(k=0;k<2;k++)

{

c[i][j]+=a[i][k]\*b[k][j];

} }}

printf("the multiply is :\n");

for(i=0;i<2;i++)

{

for(j=0;j<2;j++)

{

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

}

printf("\n");

}

break;

default:{

printf("print valid choice");

break;

}}

int var;

printf("If you want to choice again then press 1 else anything :");

scanf("%d",&var);

if(var==1)

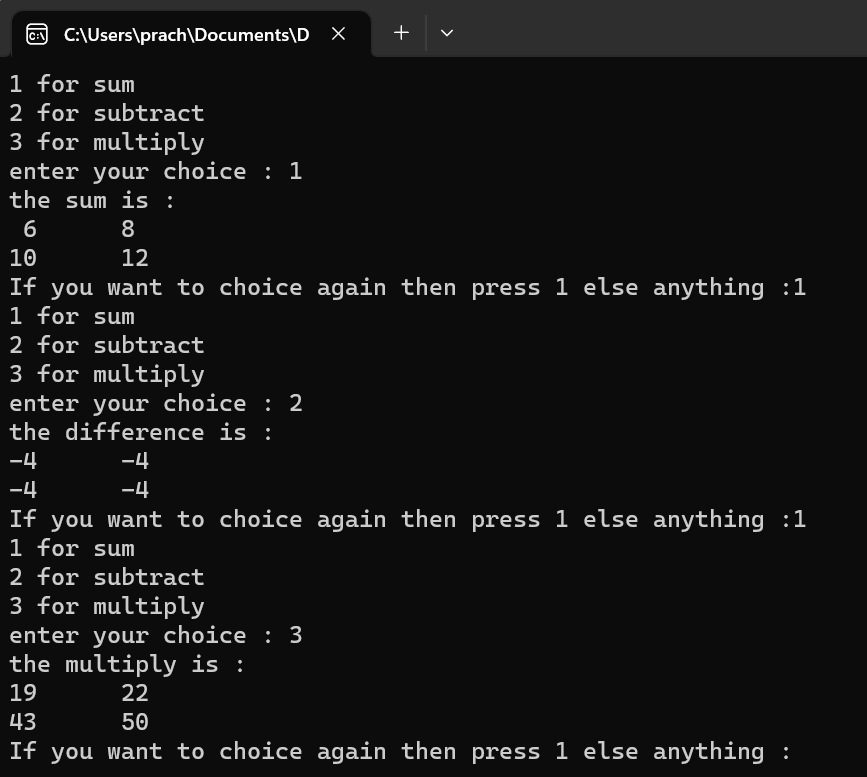
goto here;

else

return 1;

}

**Output:**

****

1. **Wap to check identity matrix**

**Source code :**

#include<stdio.h>

int main()

{

int i,j;

int r1,c1,sum=0,flag;

int a[10][10];

printf("Enter row and column of matrix : ");

scanf("%d%d",&r1,&c1);

if(r1==c1)

{

printf("\nEnter %d X %d Matrix : ",r1,c1);

for(i=0;i<r1;i++)

{

for(j=0;j<c1;j++)

{

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

}

}

for(int i=0;i<r1;i++)

{

for(int j=0;j<c1;j++)

{

if(i==j)

{

if(a[i][j]==1)

flag=1;

else

flag=0;

}

}

}

if(flag==1){

printf("matrix is identity");

}

else

printf("matrix is not identity");

}

else

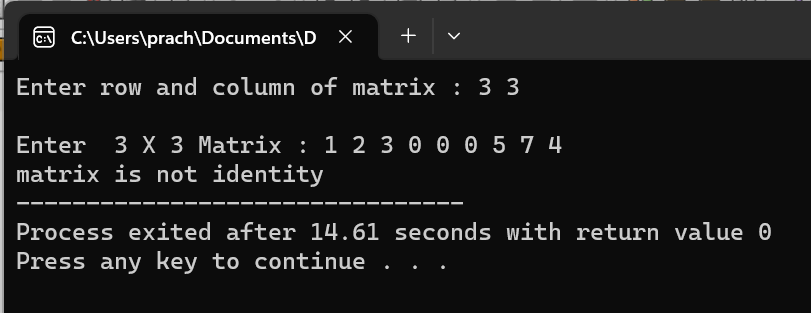
{

printf("rows and columns doesnt match");

}

}

**Output :**

****

1. **Wap check matrix is identical or not**

**Source code :**

#include<stdio.h>

int main()

{

int i,j,flag,r1,r2,c1,c2;

printf("Enter row and column of first matrix : ");

scanf("%d%d",&r1,&c1);

printf("Enter row and column of second matrix : ");

scanf("%d%d",&r2,&c2);

if(r1==r2&&c1==c2)

{

int b[r2][c2],a[r1][c1];

printf("\nEnter %d X %d Matrix : ",r1,c1);

for(int i=0;i<r1;i++)

{

for(int j=0;j<c1;j++)

{

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

}

}

printf("Enter Second %d X %d Matrix : ",r2,c2);

for(int i=0;i<r2;i++)

{

for(int j=0;j<c2;j++)

{

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

}

}

{

for(i=0;i<r1;i++)

{

for(j=0;j<c1;j++)

{

if(a[j][i]==b[i][j])

flag =0;

else

flag =1;

}

}

}

if(flag==0){

printf("matrix is identical");

}

else

printf("matrix is not identical");

}

else

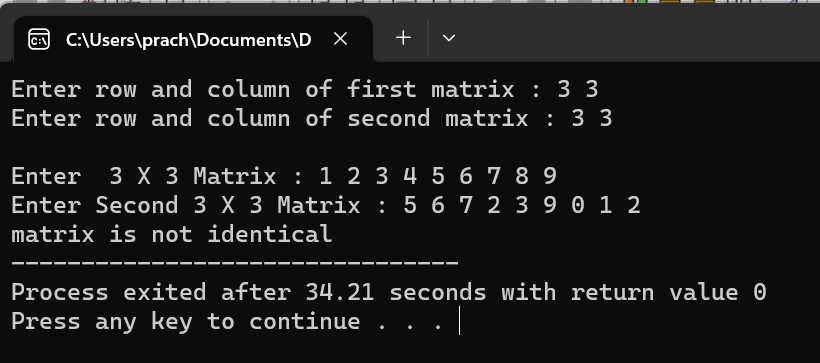
{

printf("rows and columns doesnt match");

}

}

**Output:**

****

1. **Wap to print diagonal elements.**

**Source code :**

#include<stdio.h>

int main()

{

int i,j;

int r1,c1,sum=0;

int a[10][10];

printf("Enter row and column of matrix : ");

scanf("%d%d",&r1,&c1);

printf("\nEnter %d X %d Matrix : ",r1,c1);

for(i=0;i<r1;i++)

{

for(j=0;j<c1;j++)

{

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

}

}

printf("\nThe Diagonals elements of a matrix are :: \n\n");

if(r1==c1)

{

for(i=0;i<r1;i++)

{

for(j=0;j<c1;j++)

{

if(i==j)

{

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

}

else

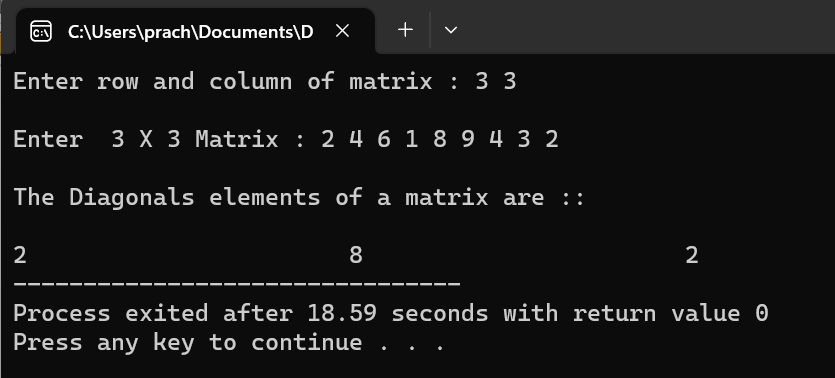
printf("\t");

}

}

}}

**Output :**

****

1. **Wap to print sum of all elements of a row**

**Source code :**

#include<stdio.h>

int main()

{ int i,j;

int r1=0,c1=0,sum=0;

printf("Enter row and column of first matrix : ");

scanf("%d%d",&r1,&c1);

printf("\nEnter %d X %d Matrix : ",r1,c1);

int a[r1][c1];

for(i=0;i<r1;i++)

{

for(j=0;j<c1;j++)

{

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

}

}

for(i=0;i<r1;i++)

{

sum = 0;

for(j=0;j<c1;j++)

{

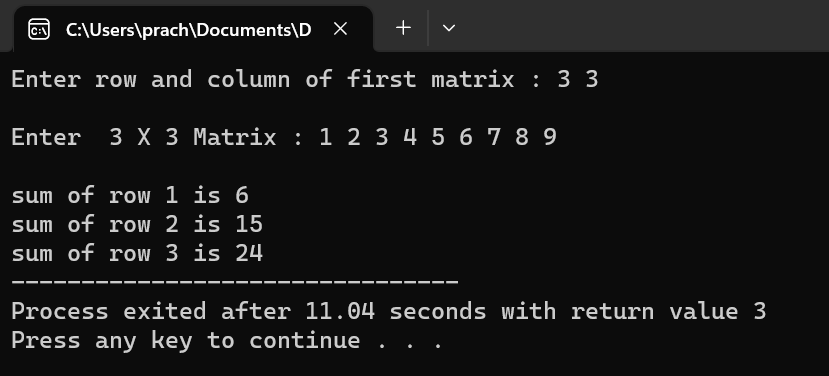
sum += a[i][j];

}

printf("\nsum of row %d is %d",i+1,sum);

}

**Output :**

****

1. **Wap to print array in row triplet form**

**Source code :**

#include<stdio.h>

int main()

{

int i,j;

int r1,c1,flag=0,k=0;

int a[10][10];

printf("Enter row and column of matrix : ");

scanf("%d%d",&r1,&c1);

for(i=0;i<r1;i++)

{

for(j=0;j<c1;j++)

{

printf("Enter the value for a[%d][%d] : ",i,j);

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

}

}

for(int i=0;i<r1;i++)

{

for(int j=0;j<c1;j++)

{

if(a[i][j]!=0)

flag++;

}

}

int m[3][flag];

for(int i=0;i<r1;i++)

{

for(int j=0;j<c1;j++)

{

if(a[i][j]!=0)

{

m[0][k]=i;

m[1][k]=j;

m[2][k]=a[i][j];

k++;

}

}

}

printf("row triplet form :\n");

{

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

{

for(int j=0;j<flag;j++)

{

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

}

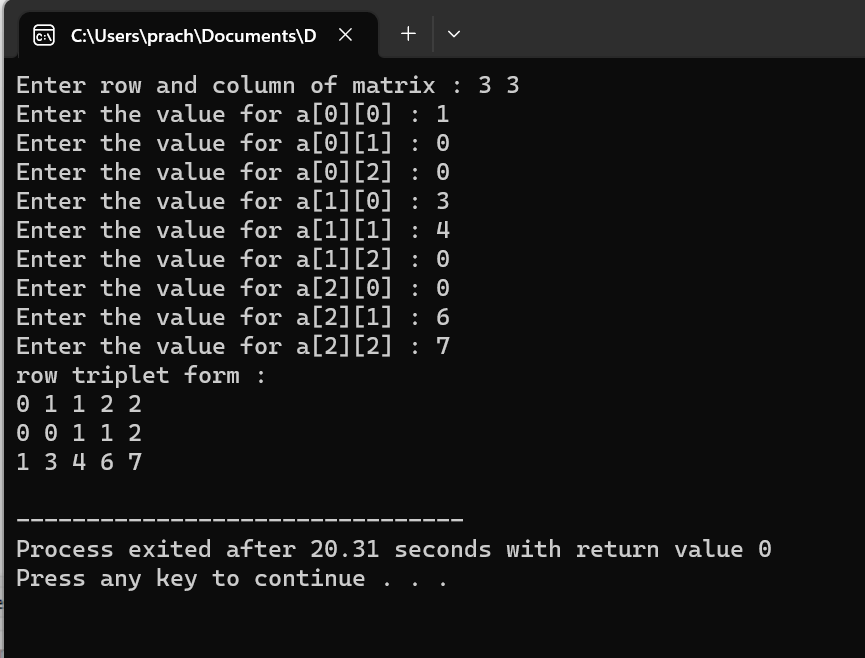
printf("\n");

}

}

}

**Output :**

****

1. **Wap to check a matrix is sparse matrix or not**

**Source code :**

#include<stdio.h>

int main()

{

int i,j;

int r1,c1,sum=0,flag=0;

int a[10][10];

printf("Enter row and column of matrix : ");

scanf("%d%d",&r1,&c1);

printf("\nEnter %d X %d Matrix : ",r1,c1);

for(i=0;i<r1;i++)

{

for(j=0;j<c1;j++)

{

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

}

}

for(int i=0;i<r1;i++)

{

for(int j=0;j<c1;j++)

{

if(a[i][j]==0)

flag++;

}

}

int temp=(r1\*c1)/2;

int temp2 = r1\*c1;

if(temp2%2 == 0)

{

if(flag>=temp)

printf("\nMatrix is sparse");

else

printf("\nMatrix is not sparce");

}else

{

if(flag>temp)

printf("\nMatrix is sparse");

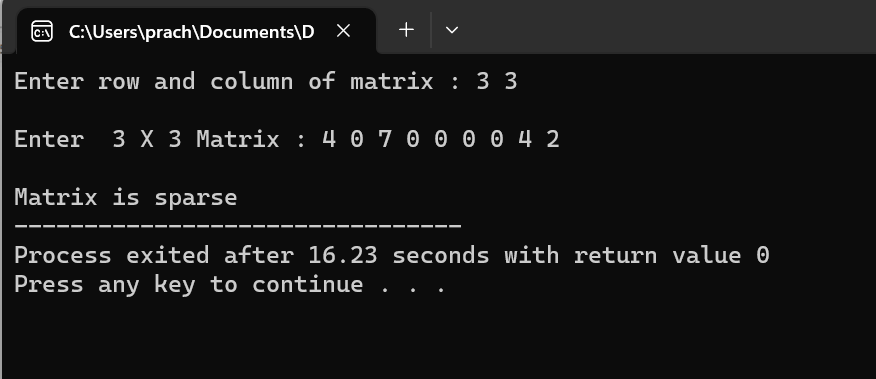
else

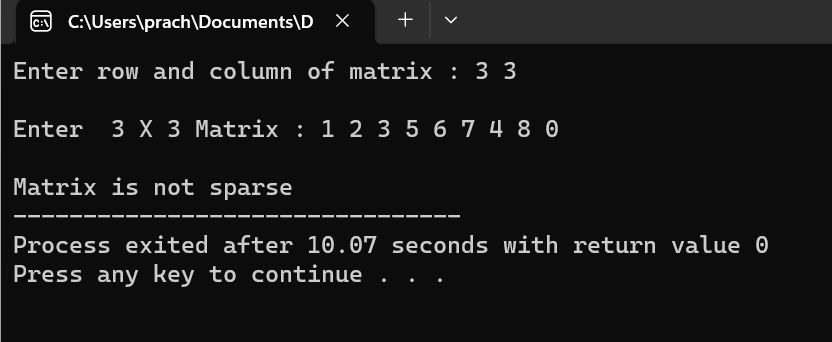
printf("\nMatrix is not sparse");

}

}

**Output:**

****



1. **Wap to arrange all rows in ascending order**

**Source code :**

#include<stdio.h>

int main(){

int arr[50][50];

int row,col,temp;

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

scanf("%d",&row);

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

scanf("%d",&col);

printf("Enter the elements: ");

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

for(int j=0;j<col;j++){

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

}

}

printf("Elements in asscendig order row wise: \n");

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

for(int j=0;j<col;j++){

for(int k=j+1;k<col;k++){

if(arr[i][j]>arr[i][k]){

temp=arr[i][j];

arr[i][j]=arr[i][k];

arr[i][k]=temp;

}

}

}

}

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

printf("Element of %d row is: ",i);

for(int j=0;j<col;j++){

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

}

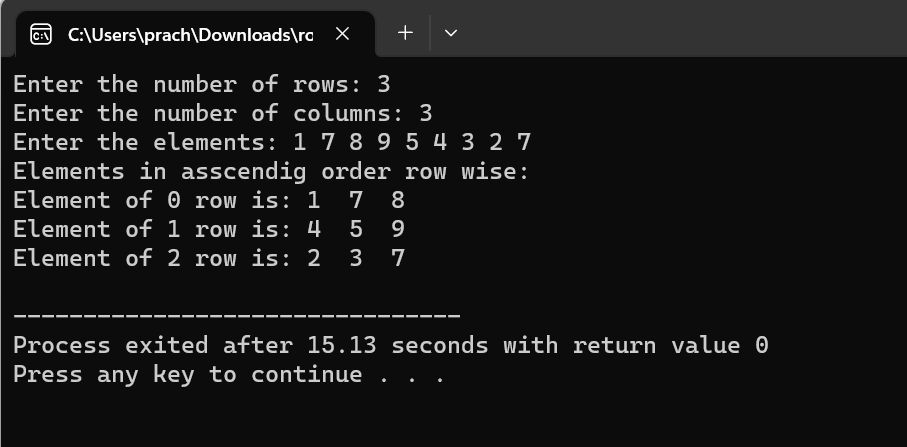
printf("\n");

}

return 0;

}

**Output :**

****

1. **Wap to print missing element in a array**

**Source code :**

#include<stdio.h>

int main()

{

int n,a[10];

printf("enter the size of array :");

scanf("%d",&n);

printf("enter array elemnts: ");

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

{

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

}

int sum=0,temp=0;

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

{

sum+=i;

}

for(int i=0;i<4;i++)

{

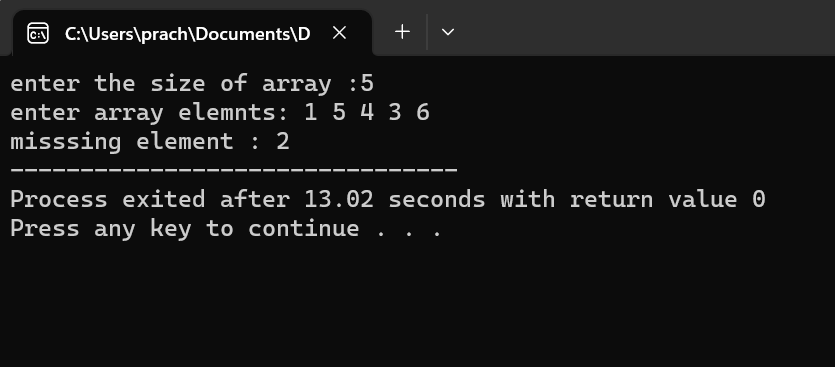
temp+=a[i];

}

printf("misssing element : %d",sum-temp);

}

**Output :**

****

1. **Wap to print duplicate element in a array**

**Source code :**

#include<stdio.h>

int main()

{

int a[10],n,count=0,flag[10];

printf("enter thye size of elements : ");

scanf("%d",&n);

printf("enter array elemnts |: ");

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

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

for(int i=0;i<n-1;i++)

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

if(a[i]==a[j])

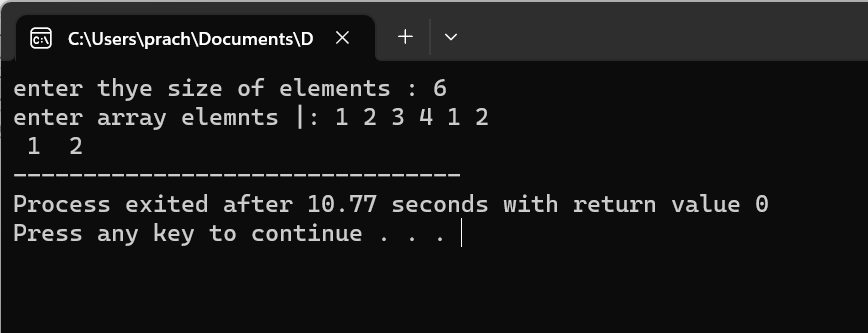
{flag[count]=a[j];

count++;}}

for(int i=0;i<count;i++)

printf(" %d ",flag[i]);}

**Output:**

****

1. **Wap to remove duplicate element from a array**

**Source code :**

#include<stdio.h>

int main()

{

int a[10],n;

printf("enter the size of array : ");

scanf("%d",&n);

printf("enter array elemnts : ");

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

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

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

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

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

for(int k=j;k<n-1;k++){

a[k]=a[k+1];

}

n--;

j--;

}  
 }

}

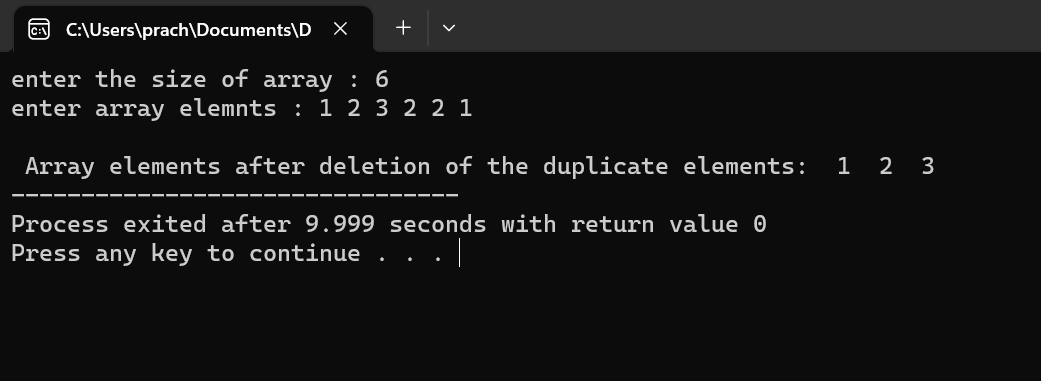
printf (" \n Array elements after deletion of the duplicate elements: ");

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

printf(" %d ", a[i]);

}

**Output:**

****

1. **Wap to create a dynamic a array and find sum using calloc**

**Source code :**

#include<stdio.h>

#include<stdlib.h>

int main()

{

int n ,\*p,sum=0;

printf("enter size of array : ");

scanf("%d",&n);

printf("Enter %d elements into the array:\n", n);

p=(int\*)calloc(n,sizeof(int));

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

scanf("%d",p+i);

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

{

sum+=\*(p+i);

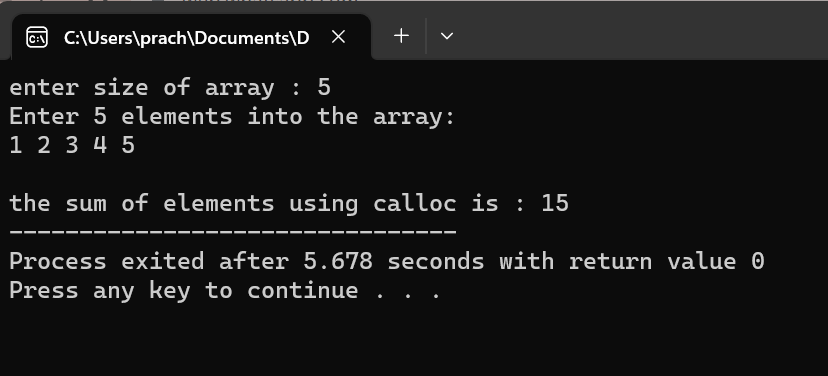
}

free(p);

printf("\nthe sum of elements using calloc is : %d",sum);

}

**Output:**

****

1. **Wap to create a dynamic a array and find sum using Malloc**

**Source code :**

#include<stdio.h>

#include<stdlib.h>

int main()

{

int n ,\*p,sum=0;

printf("enter size of array : ");

scanf("%d",&n);

printf("Enter %d elements into the array:\n", n);

p=(int\*)malloc(n\*sizeof(int));

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

scanf("%d",p+i);

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

{

sum+=\*(p+i);

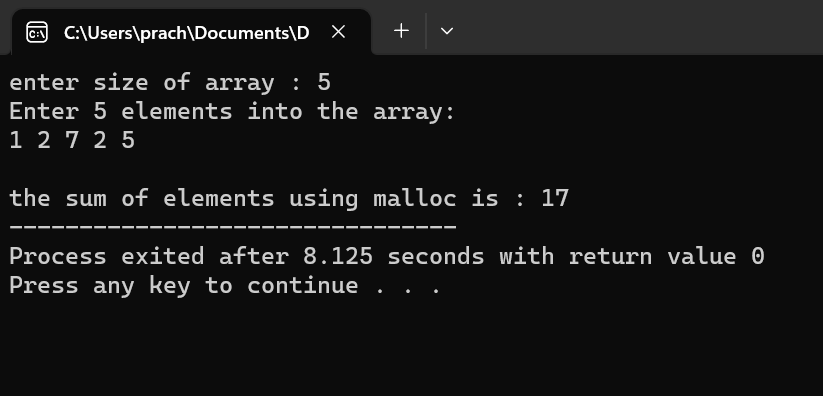
}

free(p);

printf("\nthe sum of elements using malloc is : %d",sum);

}

**Output:**

****

1. **Wap to create a dynamic a array and find sum using Realloc**

**Source code :**

#include<stdio.h>

#include<stdlib.h>

int main()

{

int n ,\*p,sum=0,size,i,j;

printf("enter size of array : ");

scanf("%d",&n);

printf("Enter %d elements into the array:\n", n);

p=(int\*)malloc(n\*sizeof(int));

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

scanf("%d",p+i);

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

sum+=\*(p+i);

printf("\nthe sum of elements before reallocation is : %d",sum);

printf("\nEnter the size you want to increase: ");

scanf("%d",&size);

p=(int \*)realloc(p,size \* sizeof(int));

printf("\nEnter new elements: ");

for(int j=i;j<n+size;j++){

scanf("%d",p+j);

}

printf("\nAll Elements: ");

for(j=0;j<n+size;j++){

printf("%d ",\*(p+j));

}

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

sum+=\*(p+j);

}

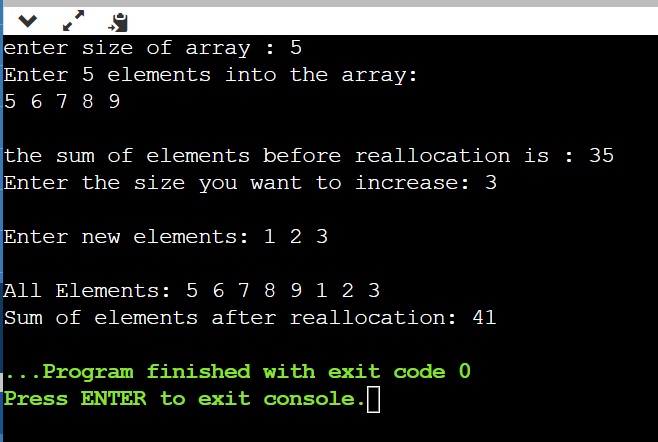
free(p);

printf("\nSum of elements after reallocation: %d",sum);

return 0;

}

**Output:**

****

1. **Wap to perform linked list operation**
2. **Insertion at Beginning**

**Source code :**

#include <stdio.h>

#include<stdlib.h>

struct node

{

int data;

struct node \*next;

};

int main()

{

int choice=1;

struct node \*new\_node;

struct node \*start=NULL;

struct node \*temp;

while(choice)

{

new\_node=(struct node\*) malloc(sizeof(struct node));

printf("enter data : ");

scanf("%d",&new\_node->data);

new\_node->next=NULL;

if(start==NULL)

{

start=new\_node; //creation of node

temp=new\_node;

}

else

{

temp->next=new\_node;

temp=new\_node;

}

printf("do you want to add new node : ");

scanf("%d",&choice);

}

temp=start;

while (temp!=NULL)

{

printf("Data is %d",temp->data); //traversinga nd printing

temp=temp->next;

}

new\_node=(struct node\*) malloc(sizeof(struct node));

printf("enter data : ");

scanf("%d",&new\_node->data);

new\_node->next=NULL;

if(start==NULL)

{

start=new\_node;

temp=new\_node;

}

else

{

new\_node->next=start;

start=new\_node;

}

printf("insertion at beginning");

temp=start;

while (temp!=NULL)

{

printf("\nData is %d",temp->data);

temp=temp->next;

}

}

**Output:**

1. **Insertion at End**

**Source code :**

#include <stdio.h>

#include<stdlib.h>

struct node

{

int data;

struct node \*next;

};

int main()

{

int choice=1;

struct node \*new\_node;

struct node \*start=NULL;

struct node \*temp;

while(choice)

{

new\_node=(struct node\*) malloc(sizeof(struct node));

printf("enter data : ");

scanf("%d",&new\_node->data);

new\_node->next=NULL;

if(start==NULL)

{

start=new\_node; //creation of node

temp=new\_node;

}

else

{

temp->next=new\_node;

temp=new\_node;

}

printf("do you want to add new node : ");

scanf("%d",&choice);

}

temp=start;

while (temp!=NULL)

{

printf("\nData is %d",temp->data); //traversing and printing

temp=temp->next;

}

new\_node=(struct node\*) malloc(sizeof(struct node));

printf("\nenter data : ");

scanf("%d",&new\_node->data);

new\_node->next=NULL;

if(start==NULL)

{

start=new\_node;

temp=new\_node;

}

else

{

temp=start;

while (temp->next!=NULL)

temp=temp->next;

}

temp->next=new\_node;

printf("insertion at end"); //inserting at the end

temp=start;

while (temp!=NULL)

{

printf("\nData is %d",temp->data);

temp=temp->next;

}

}**Output:**

1. **Insertion at Specific Position**

**Source code :**

#include<stdio.h>

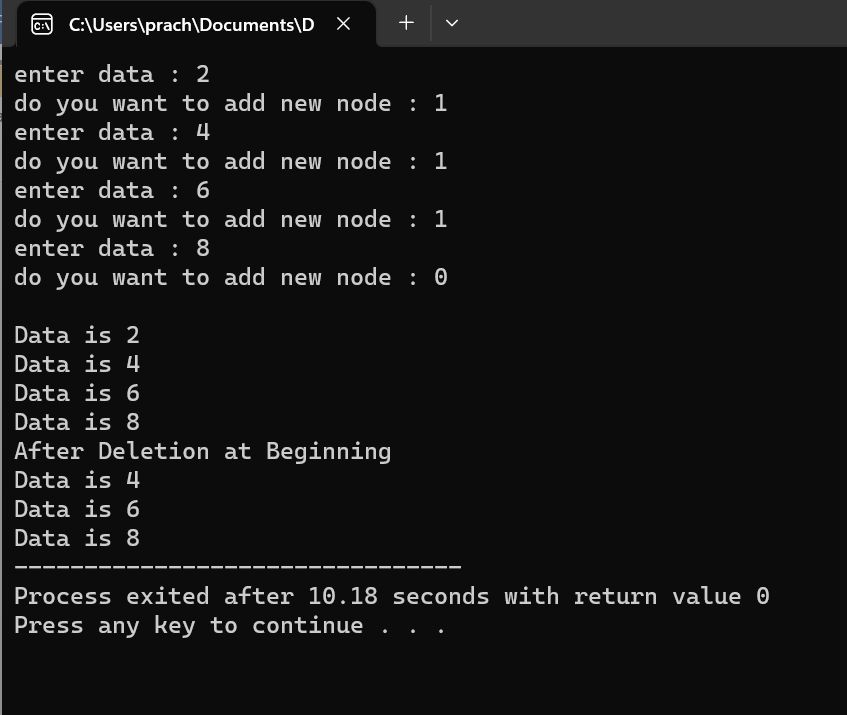
**Output:**

1. **Wap to perform linked list operation**
2. **Deletion at Beginning**

**Source code :**

#include<stdio.h>

**Output:**

****

1. **Deletion at End**

**Source code :**

#include<stdio.h>

**Output:**

1. **Deletion at Specific Position**

**Source code :**

#include<stdio.h>

**Output:**