**ASSIGNMENT-2**

# Name: Prabhnoor Singh Roll No: 102115059 Group:2NC3

**Q1. If a one dimensional integer type array with its size and length given, write the code in C++ language to create functions to perform the following operations (Please assume whatever is necessary to exemplify the results) :**

**i. Display () ii. Add/Append(x) iii. Insert (index, x) iv. Delete(x)**

**v. Linear Search(s) vi. Binary Search(x) vii. Get(index) viii. Set(index, x) ix. Max ()**

**x. Min () xi. Reverse () xii. Shift () xiii. Rotate ()**

#include <iostream> using namespace std;

void Display(int arr[],int size){

cout<<"Updated Array: "<<endl;

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

cout<<arr[i]<<endl;

}

void Add(int arr[],int size,int x){

arr[size-1]=x;

}

void Insert(int arr[],int index,int x){

arr[index-1]=x;

}

void Delete(int arr[],int size,int pos){

for(int i=pos;i<size-1;i++)

arr[i]=arr[i+1];

}

int LinearSearch(int arr[],int size,int s){ int pos=-1;

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

if(arr[i]==s){

pos=i;

break;

}

}

return pos;

}

void BinarySearch(int arr[],int size,int s){ int l,h,mid,pos; pos=-1;

l=0; h=size-1; mid=(l+h)/2;

while(mid>0){

if(arr[mid]==s){

pos=mid;

break;

}

else if(arr[mid]<s)

l=mid;

else

h=mid;

mid=(l+h)/2;

}

if(pos>=0)

cout<<s<<" is present at: "<<pos<<endl;

else

cout<<s<<" is not present"<<endl;

}

void Get(int arr[],int index){

cout<<"Number at position "<<index<<" is "<<arr[index-

1]<<endl;

}

void Set(int arr[],int index,int s){

arr[index-1]=s;

}

void Max(int arr[],int size){ int max; max=0;

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

if(arr[i]>arr[max])

max=i;

}

cout<<"Maximum is: "<<arr[max]<<" at position "<<max<<endl;

}

void Min(int arr[],int size){ int min; min=0;

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

if(arr[i]<arr[min])

min=i;

}

cout<<"Minimum is: "<<arr[min]<<" at position "<<min<<endl;

}

void Reverse(int arr[],int size){ int temp,mid; mid=size/2;

for(int i=0,j=size-1;i<mid;i++,j--){ temp=arr[i];

arr[i]=arr[j];

arr[j]=temp;

}

}

void Shift(int arr[],int size,int pos){

for(int i=size;i>pos;i--)

arr[i]=arr[i-1];

}

void Rotate(int arr[],int size,int rot,int type){ int v;

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

if(type==1){ v=arr[0];

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

arr[i]=arr[i+1];

arr[size-1]=v;

}

else if(type==0){ v=arr[size-1];

for(int i=size-1;i>0;i++)

arr[i]=arr[i-1];

arr[0]=v;

}

else{ cout<<"Invalid"<<endl;

break;

}

}

}

int main(){ int arr[100],size,index,pos,rot,type,num,ch,c; size=1; c=1;

while(c==1){ cout<<"1. DISPLAY"<<endl; cout<<"2. ADD"<<endl; cout<<"3. INSERT"<<endl; cout<<"4. DELETE"<<endl; cout<<"5. LINEAR SEARCH"<<endl; cout<<"6. BINARY SEARCH"<<endl; cout<<"7. GET"<<endl;

cout<<"8. SET"<<endl; cout<<"9. MAXIMUM"<<endl; cout<<"10. MINIMUM"<<endl; cout<<"11. REVERSE"<<endl; cout<<"12. SHIFT"<<endl; cout<<"13. ROTATE"<<endl; cout<<"ENTER YOUR CHOICE: ";

cin>>ch;

switch(ch){

case 1:

Display(arr,size);

break; case 2:

cout<<"Enter Number: "; cin>>num;

Add(arr,size,num); Display(arr,size); size=size+1;

break; case 3:

cout<<"Enter position: "; cin>>index;

cout<<"Enter Number: "; cin>>num; Shift(arr,size,index); Insert(arr,index,num); size=size+1; Display(arr,size);

break;

case 4:

cout<<"Enter Number: "; cin>>num;

pos=LinearSearch(arr,size,num); Delete(arr,size,pos); size=size-1; Display(arr,size);

break;

case 5:

cout<<"Enter Number: "; cin>>num;

pos=LinearSearch(arr,size,num);

if(pos>=0)

cout<<num<<" is present at

"<<pos<<endl;

else

cout<<num<<" is not present"<<endl;

break; case 6: cout<<"Enter Number: "; cin>>num;

BinarySearch(arr,size,num); break; case 7:

cout<<"Enter position: "; cin>>index; Get(arr,index); break; case 8:

cout<<"Enter position: "; cin>>index;

cout<<"Enter Number: "; cin>>num;

Set(arr,index,num); break; case 9:

Max(arr,size); break; case 10:

Min(arr,size); break; case 11:

Reverse(arr,size);

break;

case 12:

cout<<"Enter position: "; cin>>index;

Shift(arr,size,index); size=size+1; Display(arr,size);

break;

case 13: cout<<"Enter number of rotation: ";

cin>>rot;

cout<<"Enter 0 for Anticlockwise

Rotation"<<endl;

cout<<"Enter 1 for Clockwise Rotation"<<endl; cout<<"Choice: "; cin>>type;

Rotate(arr,size,rot,type); Display(arr,size);

break;

default:

cout<<"Invalid"<<endl;

}

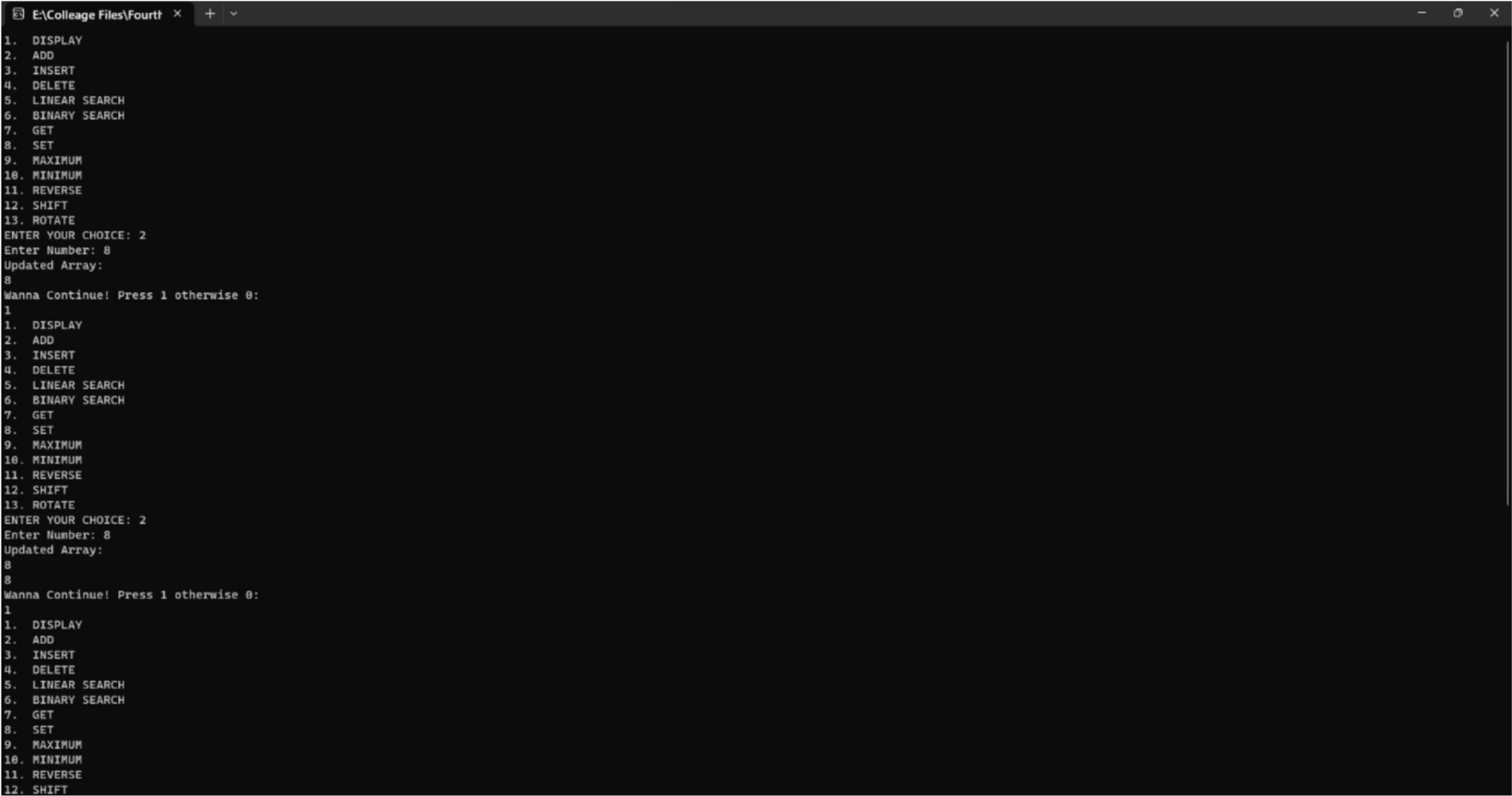
cout<<"Wanna Continue! Press 1 otherwise 0: "<<endl;

cin>>c;

}

return 0;

}



**Q2. For a given array, write functions to perform the following:**

**i. Check if an array is sorted ii. Merge arrays**

**iii. Set operations on array: Union, Intersection**

#include <iostream> using namespace std;

int arraysorted(int arr[],int size){ int flag=0; for(int i=0;i<size-1;i++){ if(arr[i]>arr[i+1]){ flag=1; break;

}

}

return flag;

}

void merge(int arr1[], int arr2[], int size1, int size2)

{

int arr3[size1+size2]; for(int i=0;i<size1;i++) arr3[i]=arr1[i]; for(int i=0;i<size2;i++) arr3[size1+i]=arr2[i]; cout<<"Merged array is :"<<endl;

for(int i=0;i<size1+size2;i++)

cout<<arr3[i]<<endl;

}

void find\_union(int arr1[],int arr2[],int size1, int size2){ int arr3[size1+size2]; int k=0,f; for(int i=0;i<size1;i++){ arr3[k]=arr1[i]; k++;

}

for(int i=0;i<size2;i++){ f=0; for(int j=0;j<size1;j++){ if(arr2[i]==arr1[j])

f=1;

}

if(f==0){ arr3[k]=arr2[i]; k++;

}

}

cout<<"Union of Arrays: "<<endl;

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

cout<<arr3[i]<<endl;

}

void find\_intersection(int arr1[],int arr2[],int size1,int size2){ int arr3[size1+size2]; int k=0,f; for(int i=0;i<size1;i++){ f=0; for(int j=0;j<size2;j++){ if(arr1[j]==arr2[i]){ arr3[k]=arr2[i]; k++;

}

}

}

cout<<"Intersection of Arrays: "<<endl;

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

cout<<arr3[i]<<endl;

}

int main(){

int size1,size2,flag;

cout<<"Size of Array1: "; cin>>size1; cout<<"Size of Array2: "; cin>>size2;

int arr1[size1],arr2[size2];

cout<<"Enter element of array 1: "<<endl;

for(int i=0;i<size1;i++) cin>>arr1[i];

cout<<"Enter element of array 2: "<<endl;

for(int i=0;i<size2;i++) cin>>arr2[i]; flag=arraysorted(arr1,size1); if(flag==1)

cout<<"Array 1 is not sorted"<<endl;

else cout<<"Array 1 is sorted"<<endl;

flag=arraysorted(arr2,size2); if(flag==1)

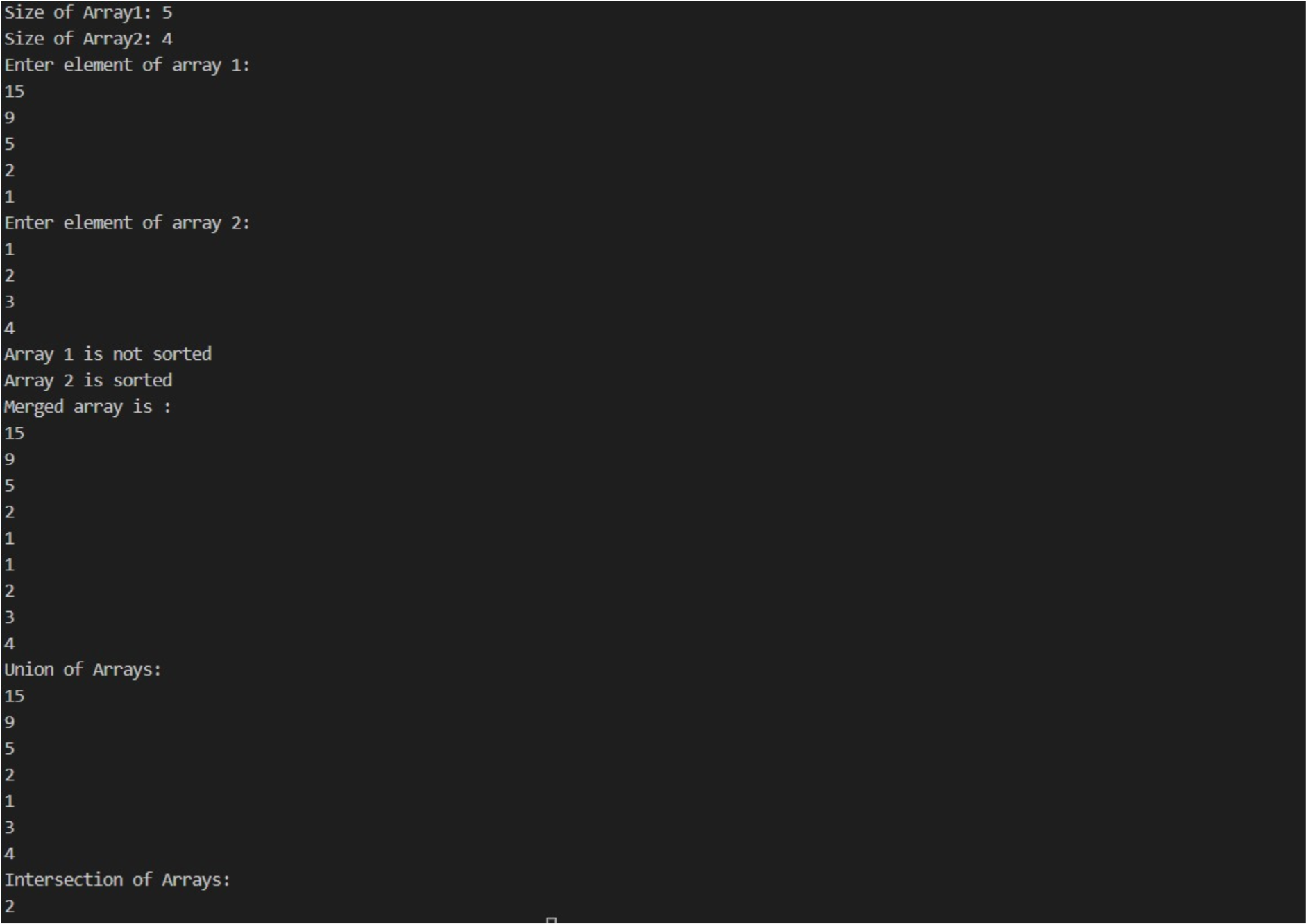
cout<<"Array 2 is not sorted"<<endl;

else cout<<"Array 2 is sorted"<<endl;

merge(arr1,arr2,size1,size2); find\_union(arr1,arr2,size1,size2); find\_intersection(arr1,arr2,size1,size2);

return 0;

}



**Q3. For a given array, write functions to perform the following:**

**i. Finding single element in an array ii. Finding multiple elements in an array iii. Finding duplicates in a sorted array iv. Finding duplicates in an unsorted array**

**v. Finding a pair of elements with sum k vi. Finding a pair of elements with sum k in sorted array vii. Finding max and min in a single scan**

#include <iostream> using namespace std;

void Ssearch(int arr[],int size, int ele){ int pos=-1;

for(int i=0;i<size;i++){ if(arr[i]==ele){

pos=i; break;

}

}

if(pos>-1)

cout<<ele<<" found at positon: "<<pos+1<<endl;

else

cout<<ele<<" not found"<<endl;

}

void Msearch(int arr[],int size,int noe){ int ele; for(int i=0;i<noe;i++){

cout<<"Enter number: ";

cin>>ele;

Ssearch(arr,size,ele);

}

}

void duplicates(int arr[],int size,int ele){//the following function works for both sorted & unsorted array int num=0;

for(int i=0;i<size;i++){ if(arr[i]==ele)

num=num+1;

}

cout<<"Number of duplicates of "<<ele<<" is: "<<num<<endl;

}

void pair\_sum(int arr[],int size,int sum){//the following function works for both sorted & unsorted array int num1=-1,num2=-1;

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

num1=i;

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

num2=j; if((num1+num2)==sum)

break;

}

}

if(num1>-1 && num2>-1) cout<<"Pair is: "<<arr[num1]<<" and "<<arr[num2]<<endl;

else

cout<<"No such pair is found"<<endl;

}

void max\_min(int arr[],int size){ int max,min; max=0; min=0;

for(int i=0;i<size;i++){ if(arr[i]>arr[max])

max=i;

if(arr[i]<arr[min])

min=i;

}

cout<<"Maximum is: "<<arr[max]<<" at position: "<<max+1<<endl; cout<<"Minimum is: "<<arr[min]<<" at position: "<<min+1<<endl;

}

int main(){ int size,ch,ele,noe,sum,c=1;

cout<<"Enter number of elements: ";

cin>>size; int arr[size];

cout<<"Enter elements: "<<endl;

for(int i=0;i<size;i++) cin>>arr[i]; while(c){

cout<<"1. Single element search"<<endl; cout<<"2. Multiple element search"<<endl; cout<<"3. Duplicate element search"<<endl; cout<<"4. Pair of elements with sum"<<endl; cout<<"5. Maximum & Minimum"<<endl;

cin>>ch; switch(ch){

case 1: cout<<"Enter element to search: ";

cin>>ele;

Ssearch(arr,size,ele); break; case 2: cout<<"Enter number of elements to search: "; cin>>noe;

Msearch(arr,size,noe); break; case 3: cout<<"Enter element: ";

cin>>ele; duplicates(arr,size,ele); break; case 4:

cout<<"Enter Sum: "; cin>>sum; pair\_sum(arr,size,sum); break; case 5: max\_min(arr,size); break; default:

cout<<"Invalid";

}

cout<<"Wanna Continue! Press 1 otherwise 0: "<<endl;

cin>>c;

}

return 0;

}

