**15B17CI371 – Data Structures Lab**

**ODD 2024**

**Week 0-LAB A**

**Practice Lab**

1.

#include <iostream>

using namespace std;

int main()

{

int n ;

cout<<"enter the number of elements in the array";

cin>>n;

int arr[n];

int sum=0;

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

{

cin>>arr[i];

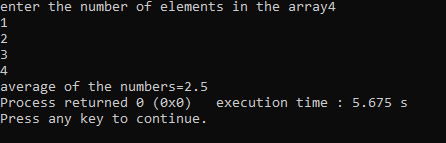
sum=sum+arr[i];

}

cout <<"average of the numbers="<<float(sum)/n;

return 0;

}



2.

#include <iostream>

using namespace std;

int main() {

int array[] = {1, 2, 3, 5, 2, 9, 7, 3, 5};

int n = sizeof(array) / sizeof(array[0]);

int unique[n];

int freq[n];

int count = 0;

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

freq[i] = 0;

}

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

int j;

for (j = 0; j < count; ++j) {

if (array[i] == unique[j]) {

freq[j]++;

break;

}

}

if (j == count) {

unique[count] = array[i];

freq[count] = 1;

count++;

}

}

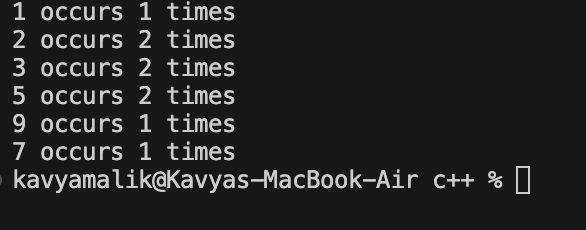
for (int i = 0; i < count; ++i) {

cout << unique[i] << " occurs " << freq[i] << " times" << endl;

}

return 0;

}



3.

#include <iostream>

using namespace std;

int main()

{

int n ;

cout<<"enter the number of elements in the array";

cin>>n;

int arr[n];

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

{

cin>>arr[i];

}

cout<<"before rotating array"<<endl;

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

{

cout<<arr[i];

cout<<" ";

}

cout<<endl<<endl;

int temp=arr[0];

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

{

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

}

arr[n-1]=temp;

cout<<"after rotating array"<<endl;

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

{

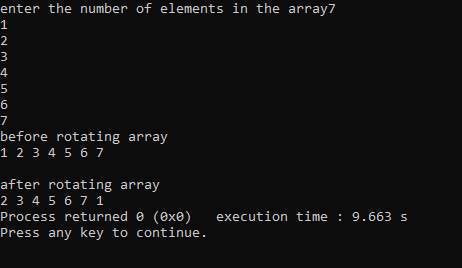
cout<<arr[i];

cout<<" ";

}

return 0;

}



4.

#include <iostream>

using namespace std;

int main()

{

int n ;

cout<<"enter the number of elements in the array";

cin>>n;

int arr[n];

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

{

cin>>arr[i];

}

cout<<endl;

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

{

cout<<arr[i];

cout<<" ";

}

cout<<endl;

int min=arr[0];

int smin=arr[1];

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

{

if(arr[i]<min)

{

smin=min;

min=arr[i];

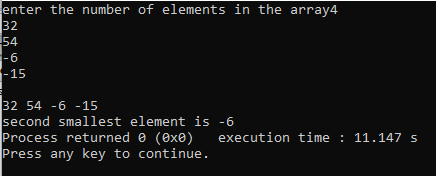
}

}

cout<<"second smallest element is "<<smin;

return 0;

}



5.

#include<iostream>

using namespace std;

int main()

{

int a;

cout<<"Enter number of inputs: ";

cin>>a;

int \*arr=new int[a];

cout<<"Enter the numbers:\n";

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

cin>>arr[i];

}

cout<<"Input: ";

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

cout<<arr[i]<<" ";

}

cout<<endl<<"Output: ";

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

if(arr[i]%2==0)

cout<<arr[i]<<" ";

}

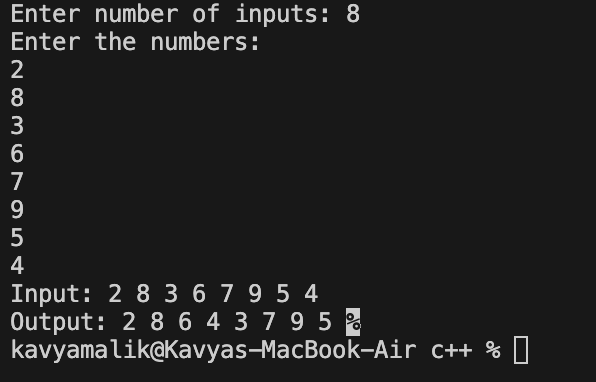
for(int i=0;i<a;i++){

if(arr[i]%2!=0)

cout<<arr[i]<<" ";

}

}



6.

#include<iostream>

using namespace std;

int main()

{

int a;

cout<<"Enter number of inputs: ";

cin>>a;

int \*A= new int[a];

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

{

A[i]=rand()%100;

}

cout<<"The randomly generated array of length "<<a<<":"<<endl;

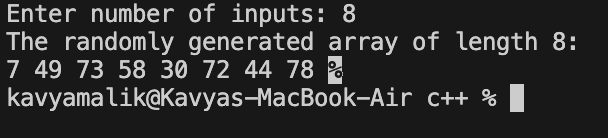
for(int i=0;i<a;i++){

cout<<A[i]<<" ";

}

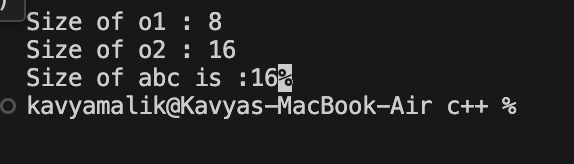
return 0;

}

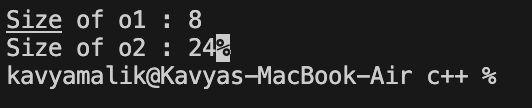


7.

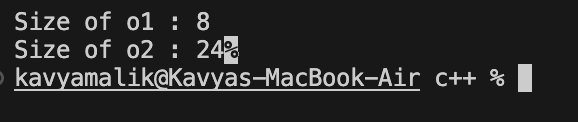
a)



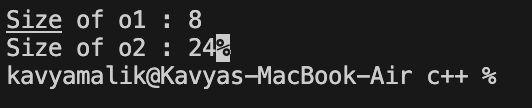
b)



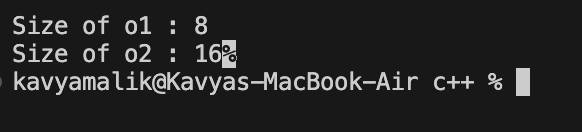
c)



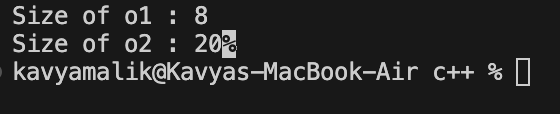
d)



e)



f)



8. Analyze the correctness and output of following programs

(a)

#include <iostream>

#include <malloc.h>

using namespace std;

int main() {

float \*a;

a = (float \*)malloc(sizeof(int));

a[0] = 4.5;

cout<<a[0];

return 0;

}

Ans : the code is correct and the output will be 4.5

(b)

#include <iostream>

#include <malloc.h>

using namespace std;

int main() {

int \*a;

a = (int \*)malloc(sizeof(float));

a[0] = 5;

cout<<a[0];

return 0;

}

Ans : the code is correct and the output will be 5

(c)

#include <iostream>

#include <malloc.h>

using namespace std;

int main() {

int \*a, \*b;

a = (int \*)malloc(sizeof(int));

b = (int \*)malloc(5\*sizeof(int));

cout<<sizeof(a)<< sizeof(b);

return 0;

}

Ans : the code is correct and the output will be 8 8.

(d)

#include <iostream>

#include <malloc.h>

using namespace std;

int main() {

int \*a;

a[0] = (int \*)malloc(sizeof(int));

a[0] = 5;

cout<<a[0];

return 0;

}

Ans : here the a is an pointer to an integer but it is uninitialized henceforth no memory can be allocated , hence accessing a[0] will lead to the termination of the code while running it.

(e)

#include <iostream>

#include <malloc.h>

using namespace std;

int main() {

int \*a[5];

a[0] = (int \*)malloc(sizeof(int));

a[0][0] = 5;

cout<<a[0][0];

return 0;

}

Ans : the code is correct and the output will be 5

(f)

#include <iostream>

#include <malloc.h>

using namespace std;

int main() {

struct node{int a[10];};

struct node \*n;

n = (struct node \*)malloc(sizeof(struct

node)); cout<<sizeof(n);

return 0;

}

Ans : int this code the part size of(struct node) calculates the size of ‘n’ rather than calculating the size of memory allocated for the struct node.

(g)

#include <iostream>

#include <malloc.h>

using namespace std;

int main() {

int \*a[5];

a[0] = (int

\*)malloc(2\*sizeof(int)); a[0][1]

= 5;

cout<<a[0][1];

return 0;

}

Ans : the code is correct and the output will be 5

(h)

#include <iostream>

#include <malloc.h>

using namespace std;

int main() {

int \*a = (int \*)malloc(5\*sizeof(int));

a[0] = 1; a[1] = 2; a[2] = 3; a[3] = 4;

a[4] = 5; delete(a);

cout<<a[0]<<a[1]<<a[2]<<a[3]<<a[4];

return 0;

}

Ans : this program will print garbage values due to the use of delete() instead of free(a).