**15B17CI371 – Data Structures Lab**

**ODD 2024**

**Week 1-LAB B**

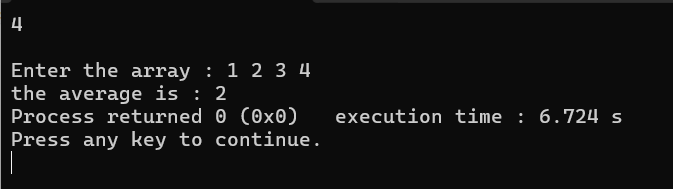
**9923103023 – Ansh Upadhyay**

**F1**

**Practice Lab**

Lab Questions:

1. Write a C/C++ program to find the average of n numbers using arrays.

ANS : 

#include<iostream>

using namespace std;

int main()

{

int n;

cin>>n;

int \*arr = new int[n];

cout<<"\nEnter the array : ";

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

{

cin>>arr[i];

}

int sum=0;

float avg;

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

{

sum+= arr[i];

}

avg = (sum)/n;

cout<<"the average is : "<<avg;

}

2. Write a C/C++ program to find the frequency of each element in an array.

Example:

Input: array = {1, 2, 3, 5, 2, 9, 7, 3, 5}

Output:

1 occurs 1 times

2 occurs 2 times

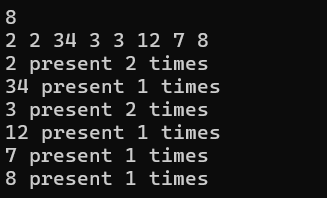
3 occurs 2 times

5 occurs 2 times

7 occurs 1 times

9 occurs 1 times

Ans :



#include<iostream>

#include<set>

using namespace std;

int main()

{

int n;

cin>>n;

int \*arr=new int[n];

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

{

cin>>arr[i];

}

set <int>s;

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

{

int cnt=0;

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

{

if(arr[i]==arr[j]) cnt++;

}

if (s.find(arr[i])==s.end()) cout<<arr[i]<<" present "<<cnt<<" times"<<endl;

s.insert(arr[i]);

}

return 0;

}

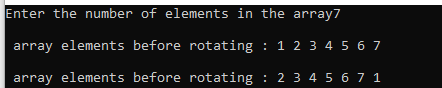
3. Given an array, write a program in C/C++to left rotate the elements of the array by one.

Example:

Array Elements before rotating: 1 2 3 4 5 6 7

Array Elements after rotating: 2 3 4 5 6 7 1

Ans :



#include<iostream>

using namespace std;

int main()

{

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

int n;

cin>>n;

int \*arr = new int[n];

cout<<"\n array elements before rotating : ";

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

{

cin>>arr[i];

}

int one=arr[0];

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

{

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

}

arr[n-1] = one;

cout<<"\n array elements before rotating : ";

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

{

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

}

}

4. Write a C/C++ program to find the second smallest element in a one-dimensional array.

Example:

Input: Array size: 4

Elements: 32 54 -6 -15

Output: -6

Ans : 

#include<iostream>

using namespace std;

int main()

{

cout<<"Input array size : ";

int n;

cin>>n;

int \*arr = new int[n];

cout<<"\n Elements : : ";

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

{

cin>>arr[i];

}

int temp=0;

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

{

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

{

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

{

temp = arr[i];

arr[i]=arr[j];

arr[j] = temp;

}

}

}

cout<<"\n the second smallest array element is : "<<arr[1];

}

5. A dynamically created array stores following integer elements (odd and even integers). It

is desired to print/display the elements of this array in such manner that it first prints all

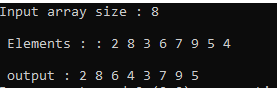
the even elements then it prints all the odd elements.

Example:

Input: 2 8 3 6 7 9 5 4

Output: 2 8 6 4 3 7 9 5

Ans :



#include<iostream>

using namespace std;

int main()

{

cout<<"Input array size : ";

int n;

cin>>n;

int \*arr = new int[n];

cout<<"\n Elements : : ";

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

{

cin>>arr[i];

}

cout<<"\n output : ";

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

{

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

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

}

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

{

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

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

}

}

6. Write a program without STL to create the dynamic array of user inputted length (n),

assign values at different indices of the array, and as presented in above example,

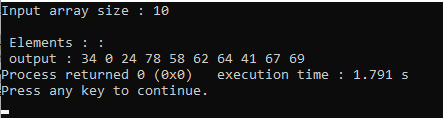
display the elements of this array.

(Note: don’t enter the elements manually, rather use following statement in loop to

randomly assign elements (in range between 0 and 99) in the array:

A[i] = rand()%100, where A is an array).

Ans :



#include<iostream>

#include<bits/stdc++.h>

using namespace std;

int main()

{

cout<<"Input array size : ";

int n;

cin>>n;

int \*arr = new int[n];

cout<<"\n Elements : : ";

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

{

arr[i] = rand()%100;

}

cout<<"\n output : ";

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

{

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

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

}

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

{

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

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

}

}

7. Considering that an integer variable, a float variable, a double variable, a character

variable, and a pointer variable need 4, 4, 8, 1, and 8 bytes memory space respectively,

what will be the output of following C++ programs.

(a)

#include <iostream>

using namespace std;

class abc

{

int x;

double y;

};

int main()

{

abc \*o1 = new abc, o2;

cout<<"\nSize of o1 : "<<sizeof(o1);

cout<<"\nSize of o2 : "<<sizeof(o2);

cout<<"\nSize of abc is :"<<sizeof(abc);

return 0;

}

Ans : Size of o1 : 4

Size of o2 : 16

Size of abc is :16

(b)

#include <iostream>

using namespace std;

class abc

{

int x;

double y;

int z;

};

int main()

{

abc \*o1 = new abc, o2;

cout<<"\nSize of o1 : "<<sizeof(o1);

cout<<"\nSize of o2 : "<<sizeof(o2);

return 0;

}

Ans :

Size of o1 : 4

Size of o2 : 24

(c)

#include <iostream>

using namespace std;

class abc

{

int x;

double y;

int z;

int a;

};

int main()

{

abc \*o1 = new abc, o2;

cout<<"\nSize of o1 : "<<sizeof(o1);

cout<<"\nSize of o2 : "<<sizeof(o2); return 0;

}

Ans : Size of o1 : 4

Size of o2 : 24

(d)

#include <iostream>

using namespace std;

class abc

{

float x;

char y;

int z;

double a;

};

int main()

{

abc \*o1 = new abc, o2;

cout<<"\nSize of o1 : "<<sizeof(o1);

cout<<"\nSize of o2 : "<<sizeof(o2);

return 0;

}

Ans : Size of o1 : 4

Size of o2 : 24

(e)

#include <iostream>

using namespace std;

class abc

{

char x[5];

double y;

};

int main()

{

abc \*o1 = new abc, o2;

cout<<"\nSize of o1 : "<<sizeof(o1);

cout<<"\nSize of o2 : "<<sizeof(o2); return 0;

}

Ans : Size of o1 : 4

Size of o2 : 16

(f)

#include <iostream>

using namespace std;

class abc

{

char x[5];

float y[3];

};

int main()

{

abc \*o1 = new abc, o2;

cout<<"\nSize of o1 : "<<sizeof(o1);

cout<<"\nSize of o2 : "<<sizeof(o2);

return 0;

}

Ans : Size of o1 : 4

Size of o2 : 20

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 4 4

(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).