MINISTRY OF EDUCATION AND SCIENCE OF UKRAINE NATIONAL TECHNICAL UNIVERSITY

“KHARKIV POLYTECHNIC INSTITUTE”

DEPARTMENT OF SOFTWARE ENGINEERING AND MANAGEMENT INFORMATION TECHNOLOGIES

### PROGRAMING BASICS

### Laboratory Training 4

# Use of Arrays and Pointers

Executed by: SALEM CHAOUFI of group КН-221 iв.e

KHARKIV 2021

## 1 Training Tasks

### 1.1 Binary Representation

Write a program that reads unsigned short integers and prints their binary representation. Left-hand zeros should not be printed.

#include <iostream>

using namespace std;

int main()

{

int a[10], n, i;

cout<<"Enter the number to convert: ";

cin>>n;

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

{

a[i]=n%2;

n= n/2;

}

cout<<"Binary of the given number= ";

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

{

cout<<a[i];

}

}



### 1.2 Sum of the Minimum and Maximum Items

Write a program that calculates the sum of the minimum and maximum items of an array of double precision floating point values. Use two separate functions.

#include <iostream>

#include <math.h>

#include <cmath>

using namespace std;

void maxmin(double \*array, int n) {

static double max;

static double min;

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

cout << "input number " << i + 1 << " ; " << endl;

cin >> array[i];

if (i == 0) {

min = array[i];

max = array[i];

}

else {

if (max < array[i]) {

max = array[i];

}

if (min > array[i]) {

min = array[i];

}

if (i + 1 == n) {

cout << "the max value; " << max << " . the min value; " << min << endl;

cout << "sum of max and min is; ";

cout << max + min << endl;

}

}

}

}

void result() {

int i = 0;

double n;

cout << "input how much numbers you want; ";

cin >> n;

double\* array = new double[n];

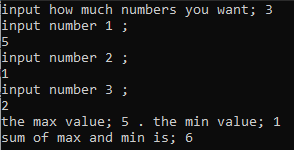
maxmin(array, n);

}

int main() {

result();

}



### 1.3 Descending Order

Write a program that sorts items of an array of integers in descending order.

#include <iostream>

using namespace std;

int main()

{

int array\_size;

cout << "input how much elements you want; ";

cin >> array\_size;

int\* array = new int[array\_size];

for (int i = 0; i < array\_size; i++) {

cout << "input element " << i + 1 << " ; " << endl;

cin >> array[i];

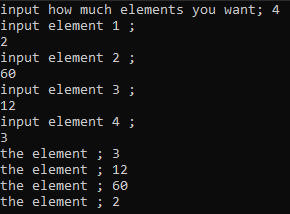
}

for (int j = array\_size - 1; j >= 0; j--) {

cout << "the element ; " << array[j] << endl;

}

}



### 1.4 Array in Free Store

Write a program that reads from keyboard the size of a two-dimensional array, allocates an array in free store, reads array items from keyboard, calculates sums of rows and puts these sums into a new array.

#include <iostream>

using namespace std;

int main()

{

int sum ;

int r ;

cout << "input how much rows you want: " ;

cin >> r ;

int c ;

cout << "input how much colomns you want: " ;

cin >> c ;

int array\_rows = r ;

int array\_colomns = c ;

int \*\*array;

array = new int\*[r];

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

cout << " \* input colomns of row " << i+1 << " : " << endl ;

array[i]=new int[i];

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

cout << "input column " << j << " : " << endl ;

cin >> array[i][j] ;

}

}

for(int s=0 ; s<r ; s++ ){

sum = 0 ;

// array[s]=new int[s];

cout << endl ;

cout << "sum of rows " << s+1 << " : " << endl ;

for (int v=0 ; v<c ; v++ ){

cout << array[s][v] << " " << endl ;

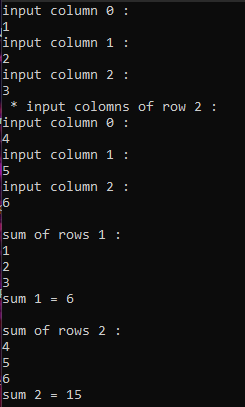
sum += array[s][v] ;

}

cout << "sum " << s+1 << " = " << sum << endl ;

}

}



### 1.5 Individual Assignment

You should create a program that defines and initializes a two-dimensional array of *integer* items and then implements following activities:

* transformation of the source array according to step one of the individual assignment
* creation and filling of a new (one-dimensional) array of *double precision floating point* type items according to step two of the individual assignment
* output of both array items

You should create one-dimensional array in free store (heap) using **new** operator and remove it before end of execution using **delete** operator. The program should signal errors if transformation or filling are not possible.

#include <iostream>

#include <math.h>

#include <cmath>

using namespace std;

int main()

{

int array\_rows = 4;

int array\_column = 3;

int i = array\_rows;

int j = array\_column;

int \*\*array;

double \*arr;

array = new int \*[4];

arr = new double[4];

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

{

array[i] = new int[i];

cout << " Input row " << i + 1 << " : " << endl;

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

{

cout << "Input column " << j + 1 << " : ";

cin >> array[i][j];

}

}

cout << " All items with odd values should be doubled : " << endl;

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

{

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

{

if (array[s][v] % 2 != 0)

{

int d = array[s][v] \* 2;

cout << "item : " << d << endl;

}

}

}

cout << " Square roots of minimal positive items of rows : " << endl;

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

{

int min;

int max;

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

{

if (array[s][v] > 0)

{

if (v == 0)

{

max = array[s][v];

min = array[s][v];

}

if (max < array[s][v])

{

max = array[s][v];

}

if (min > array[s][v])

{

min = array[s][v];

}

}

}

cout << "row : " << s + 1 << endl;

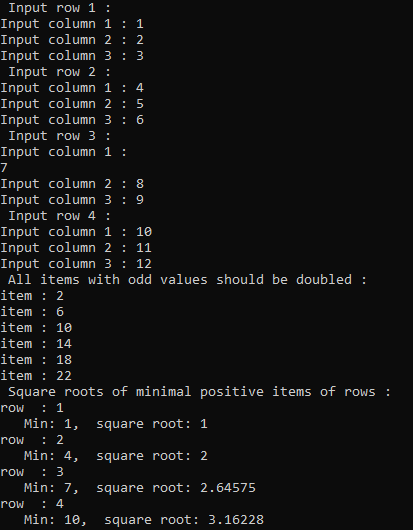
cout << " Min: " << min << ", square root: " << sqrt(min) << endl;

arr[s] = sqrt(min);

}

delete array, arr;

}



CONCLUSION :

Because of this lab, now I know how to deal with arrays and pointers of all kinds in c++.