Algorithms and Data Structures

laboratory work 2

Abakirov Nursultan

SE (english) 1-16

**Контрольные вопросы**

1. Чем отличаются статические и динамические величины?

Статические велиины не подлежать

1. Какая память называется динамически распределяемой?
2. Какой оператор используется для получения адреса переменной?

Амперсант &

1. Какой оператор позволяет получить значение по адресу, содержащемуся в указателе?
2. Что такое указатель?
3. Приведите примеры объявления указателей

int \*pointer = new int;

1. В чем различие между адресом хранящимся в указателе, и значением записанным по этому адресу?
2. Как выделить память под динамическую переменную? Как освободить память от динамической переменной?
3. Что в языке C++ означает константа NULL?
4. В чем различие между оператором разыменования и оператором получения адреса?
5. В чем различие межу объявлениями

*Const int \*ptr1* и

*int const ptr2*

1. В каком случае возможно присваивание указателей?
2. Что такое утечка памяти?
3. Какие ситуации приводят к возникновению в динамически распределяемой памяти "мусора"?
4. В чем разница между ссылкой и указателем?
5. Когда нужно использовать указатель, а не ссылку?

TASK 1

1. Problem statement:

To find the minimum element of the array (5) and its sequence number

2. Input and Output data

input data:

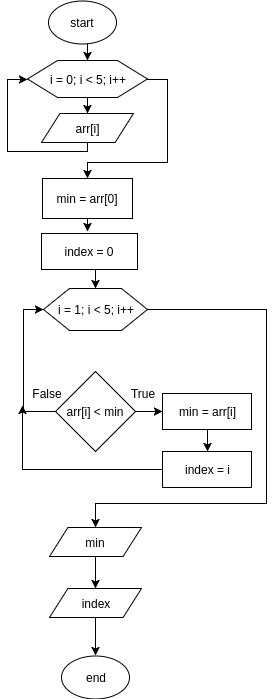
arr[5] – array of real numbers

output data:

min – minimum element of the array

index – index of minimum element of the array

3. Algorithm

#include <iostream>

using namespace std;

int main(){

cout << "input 5 elements of array: ";

float arr[4];

for (short i = 0; i < 5; i++) {

cin >> arr[i];

}

float min = arr[0];

short index = 0;

for (short i = 1; i < 5; i++) {

if (arr[i] < min) {

min = arr[i];

index = i;

}

}

cout << "min el: " << min << endl;

cout << "index: " << index;

cout << endl;

return 0;

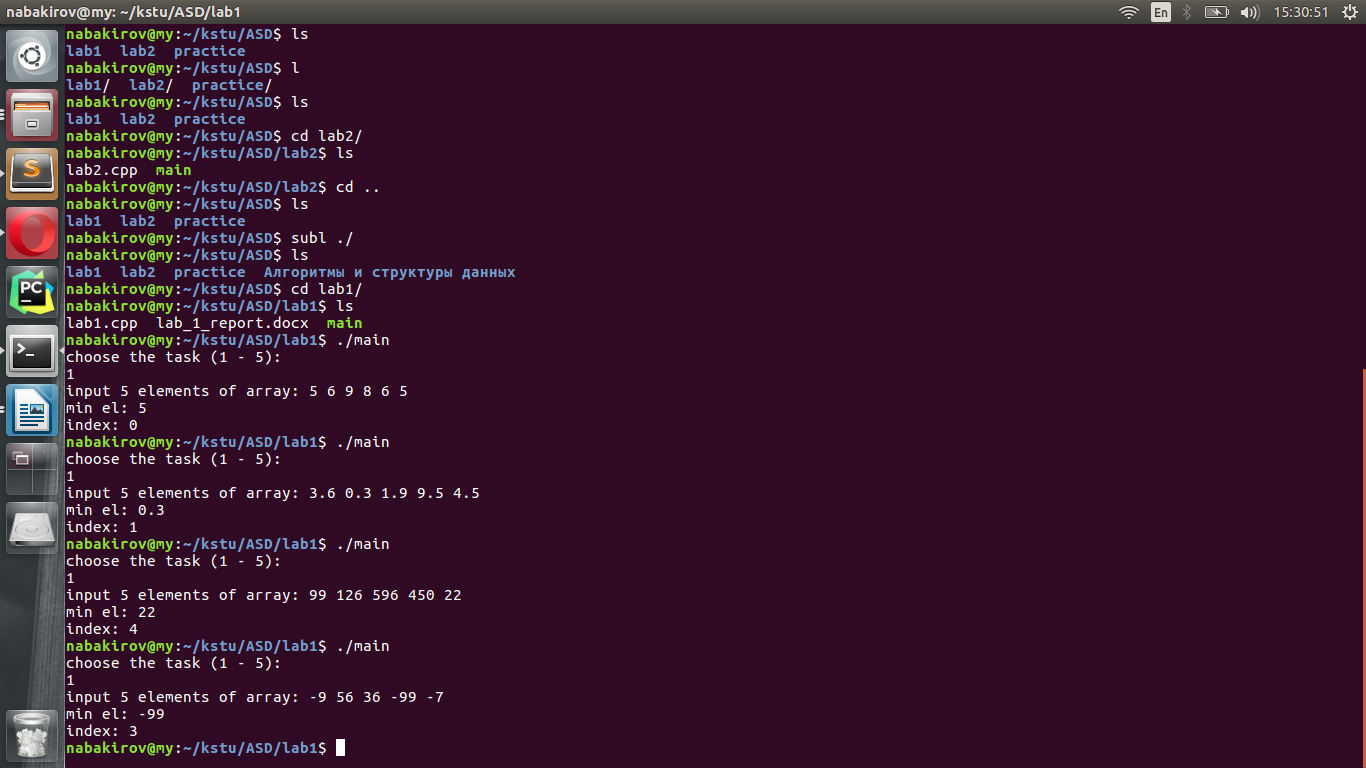
}

Tests

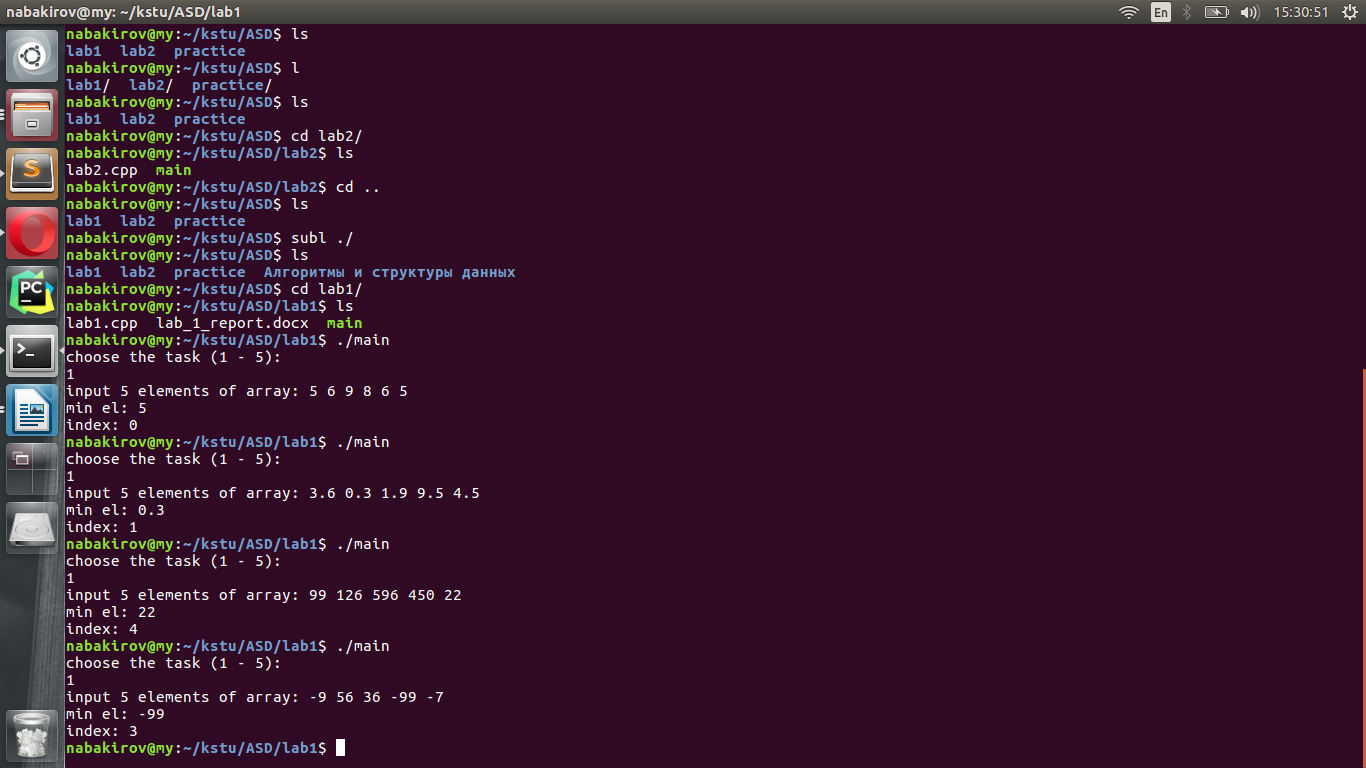
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Test № | arr[0] | arr[1] | arr[2] | arr[3] | arr[4] | min | index |
| 1 | 5 | 6 | 9 | 8 | 6 | 5 | 0 |
| 2 | 3.6 | 0.3 | 1.9 | 9.5 | 4.5 | 0.3 | 1 |
| 3 | 99 | 126 | 596 | 450 | 22 | 22 | 4 |
| 4 | -9 | 56 | 36 | -99 | -7 | -99 | 3 |

Proof:

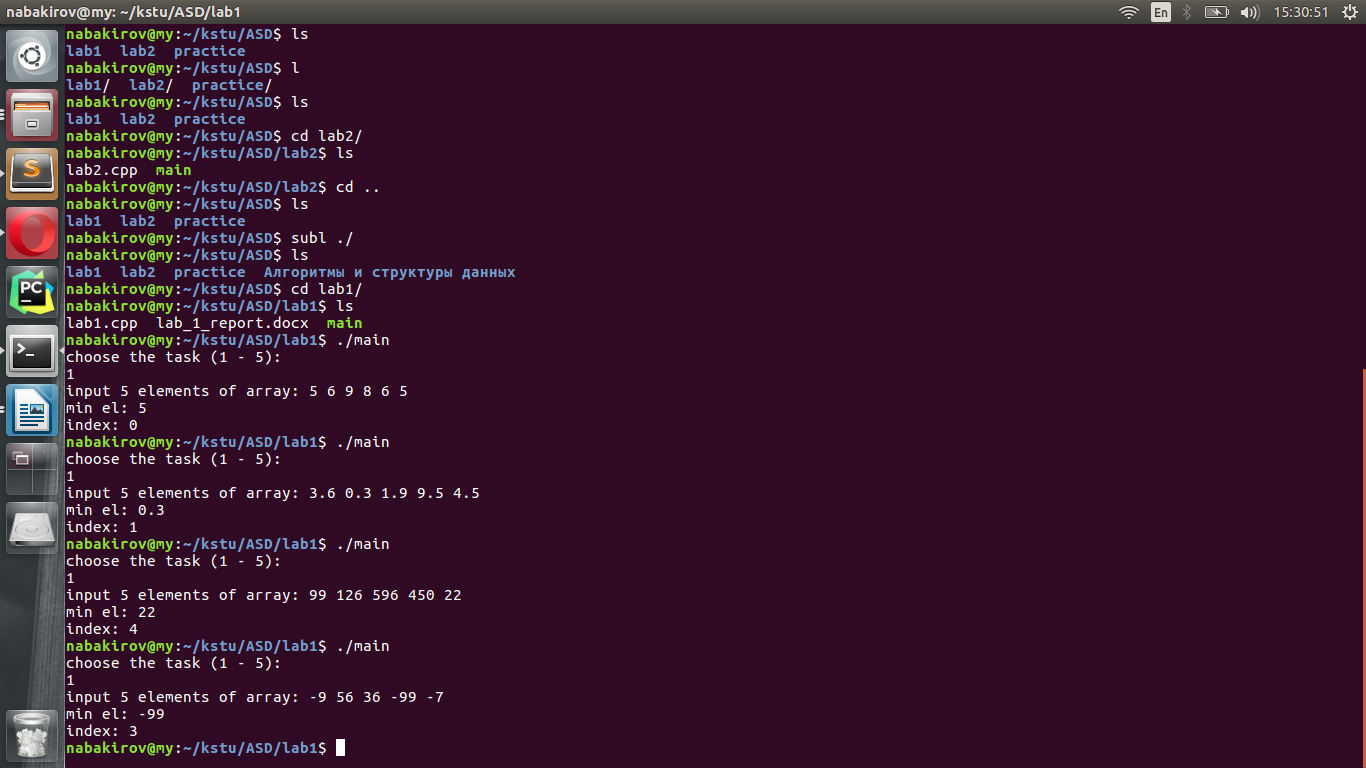
1



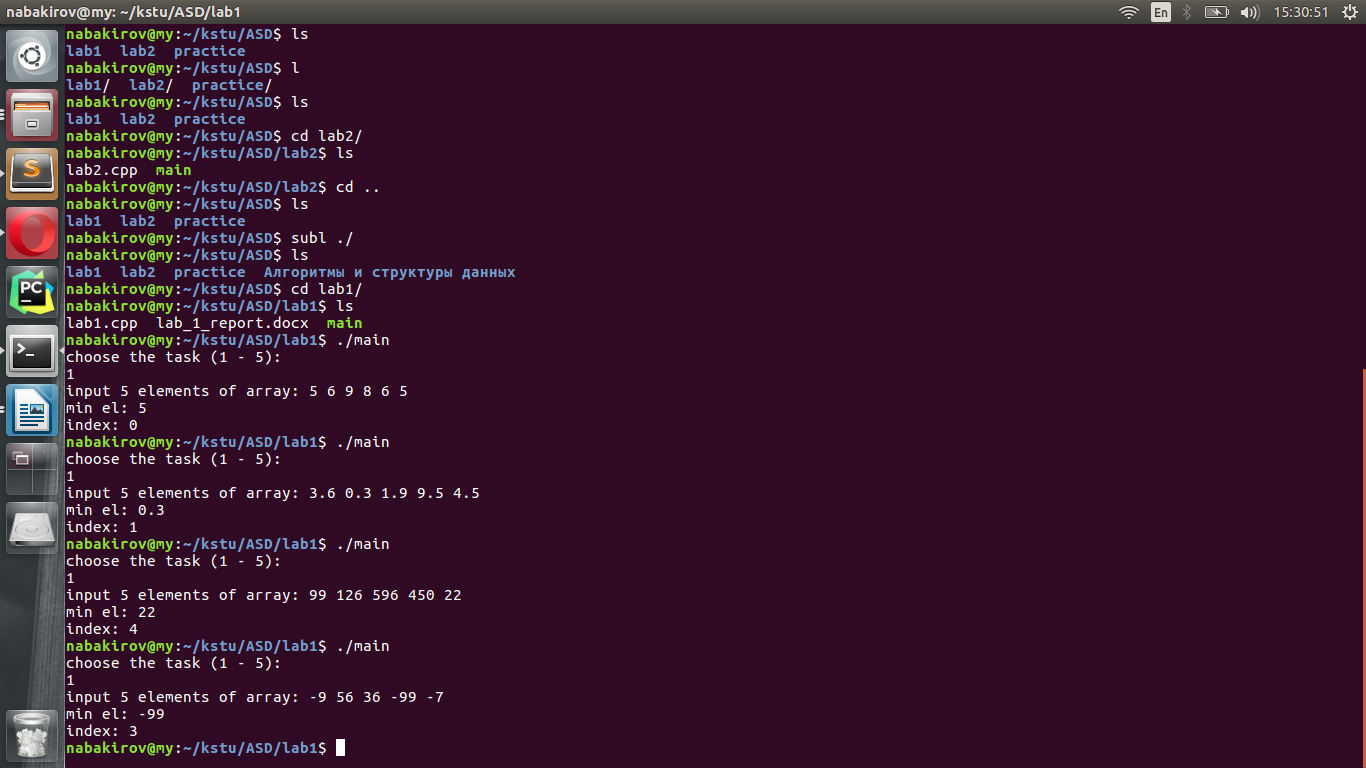
2



3



4



Task 2

Lab work

1. Problem statement:

To find quantity of elements in the array (A = 10), situated after the smallest element

2. Input and Output data

input data:

arr[10] – array of real numbers

output data:

index – integer, index of the smallest element

quantity – integer, quantity of number situated after the smallest element

3. Algorithm

#include <iostream>

using namespace std;

int main() {

cout << "input 10 elements of array: ";

float arr[10];

for (short i = 0; i < 10; i++) {

cin >> arr[i];

}

float min = arr[0];

short index = 0;

for (short i = 1; i < 10; i++) {

if (arr[i] <= min) {

min = arr[i];

index = i;

}

}

int result = 10 - (index + 1);

cout << "index of min: " << index << endl;

cout << "quantity: "<< result<< endl;

return 0;

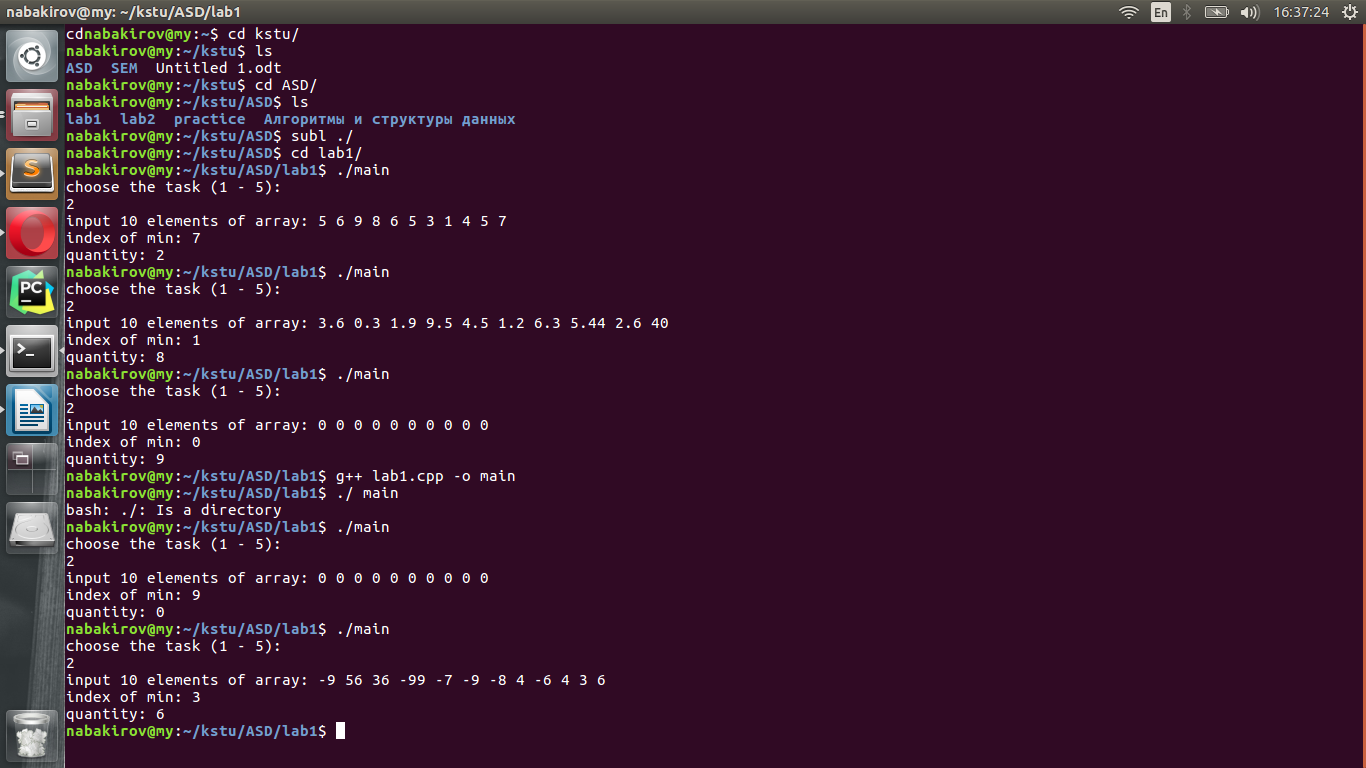
}

Tests

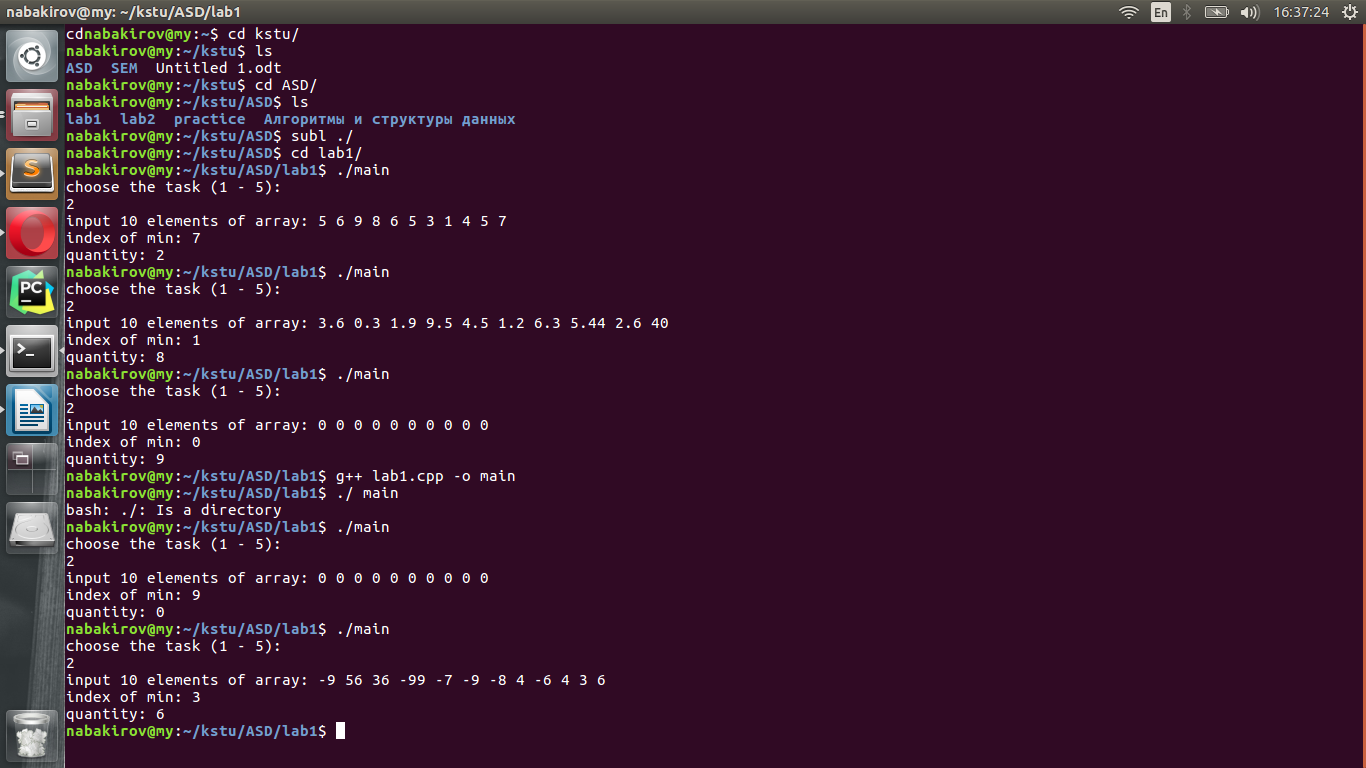
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Test № | arr[0] | arr[1] | arr[2] | arr[3] | arr[4] | arr[5] | arr[6] | arr[7] | arr[8] | arr[9] | Min  index | quantity |
| 1 | 5 | 6 | 9 | 8 | 6 | 5 | 3 | 1 | 4 | 5 | 7 | 2 |
| 2 | 3.6 | 0.3 | 1.9 | 9.5 | 4.5 | 1.2 | 6.3 | 5.4 | 2.6 | 40 | 1 | 8 |
| 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 9 | 0 |
| 4 | -9 | 56 | 36 | -99 | -7 | -9 | -8 | 4 | -6 | 4 | 3 | 6 |

Proof:

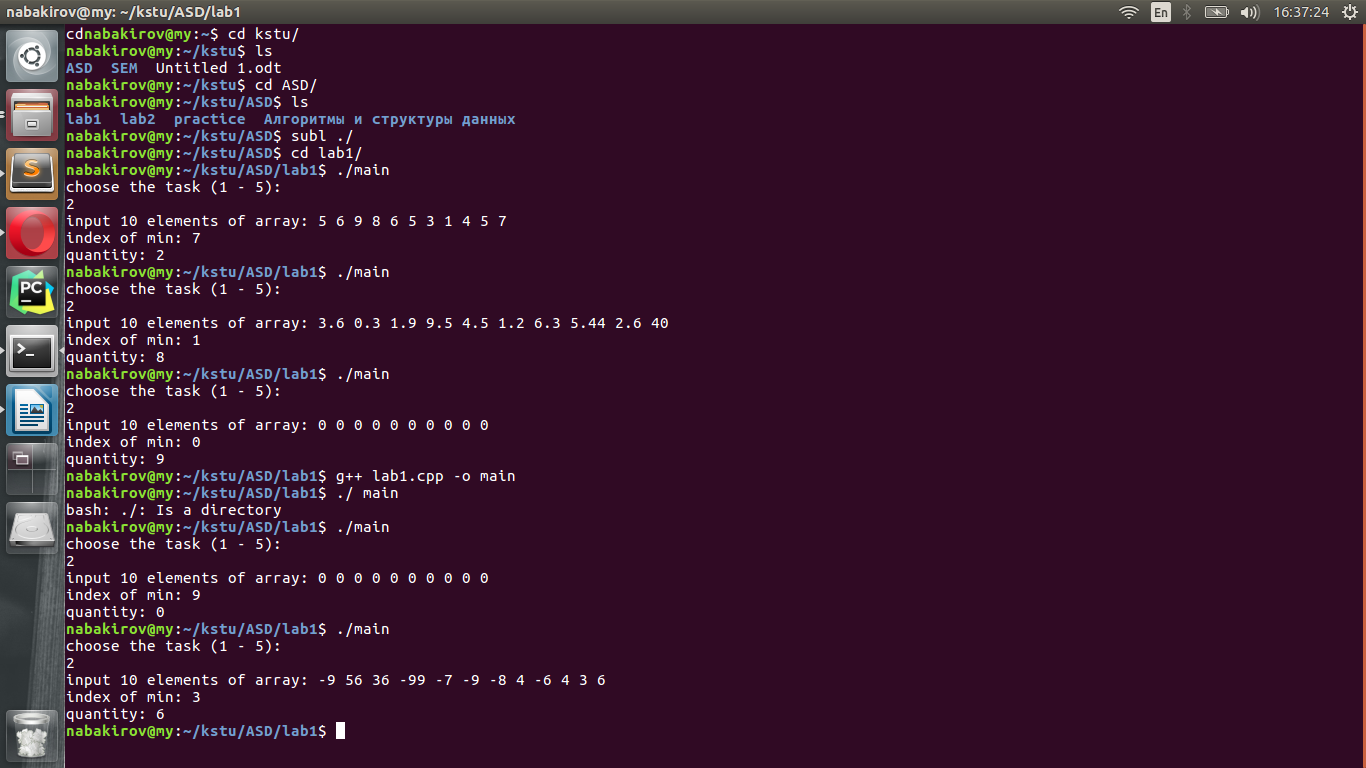
1



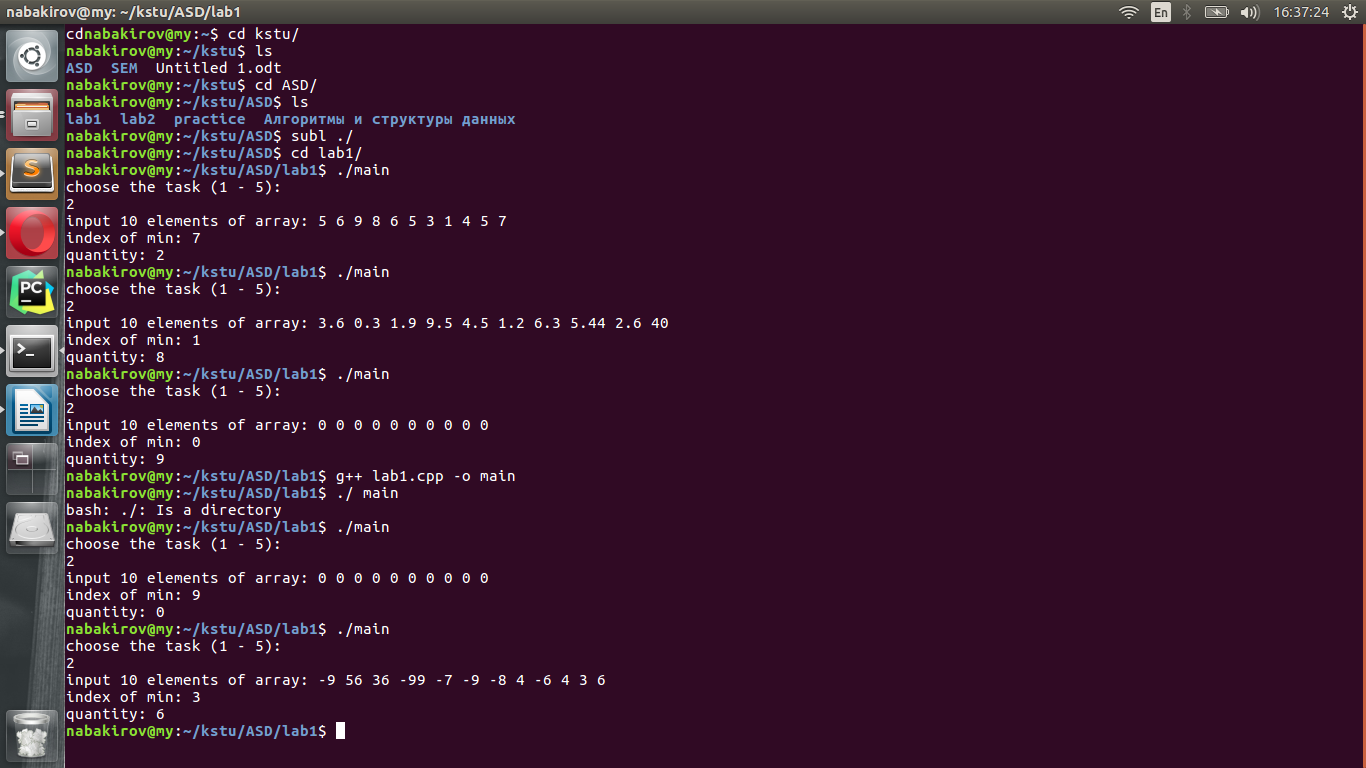
2



3



4



Task 3

1. Problem statement:

To swap lines containing a minimal element and a maximal element in the specified 2-dimensional array (A = 3x3)

2. Input and Output data

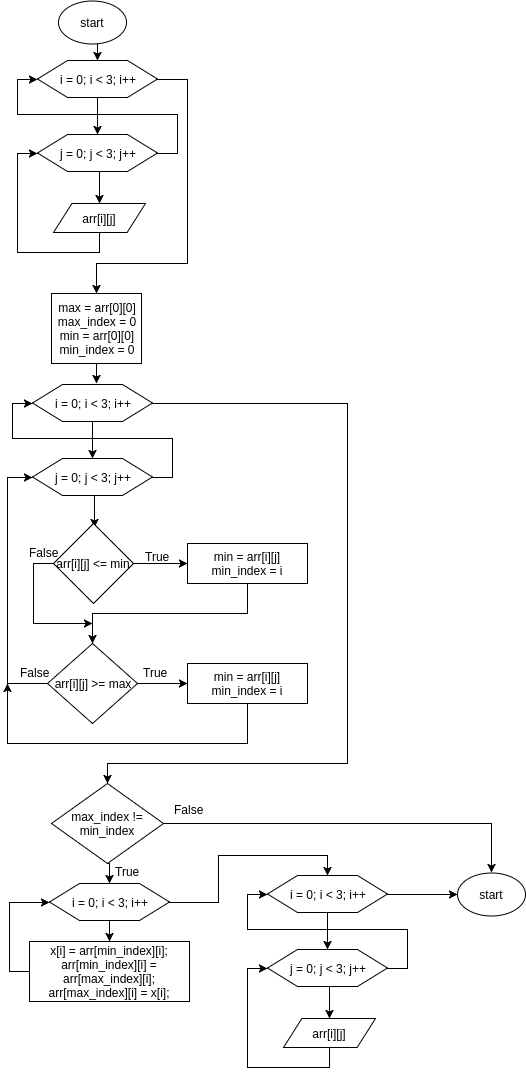
input data:

arr[3][3 – 2-dimensional array of real numbers

output data:

arr[3][3] – 2-dimensional array of real numbers

3. Algorithm



include <iostream>

using namespace std;

int main() {

cout << "input 9 elements of 2 dimentional array: ";

float arr[3][3];

for(short i = 0; i < 3; i++){

for(short j = 0; j < 3; j++){

cin >> arr[i][j];}}

float x[3];

float min = arr[0][0];

short min\_index = 0;

float max = arr[0][0];

short max\_index = 0;

for(short i = 0; i < 3; i++){

for(short j = 0; j < 3; j++){

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

min = arr[i][j];

min\_index = i;}

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

max = arr[i][j];

max\_index = i;}}}

if (max\_index != min\_index){

for (short i = 0; i < 3; i++){

x[i] = arr[min\_index][i];

arr[min\_index][i] = arr[max\_index][i];

arr[max\_index][i] = x[i];}}

cout << endl;

for(short i = 0; i < 3; i++){

for(short j = 0; j < 3; j++){

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

cout << endl;}}

Tests

input: Input: Input:

1 2 6 0 0 0 -34 -21 -56

4 8 9 4 8 7 98 56 87

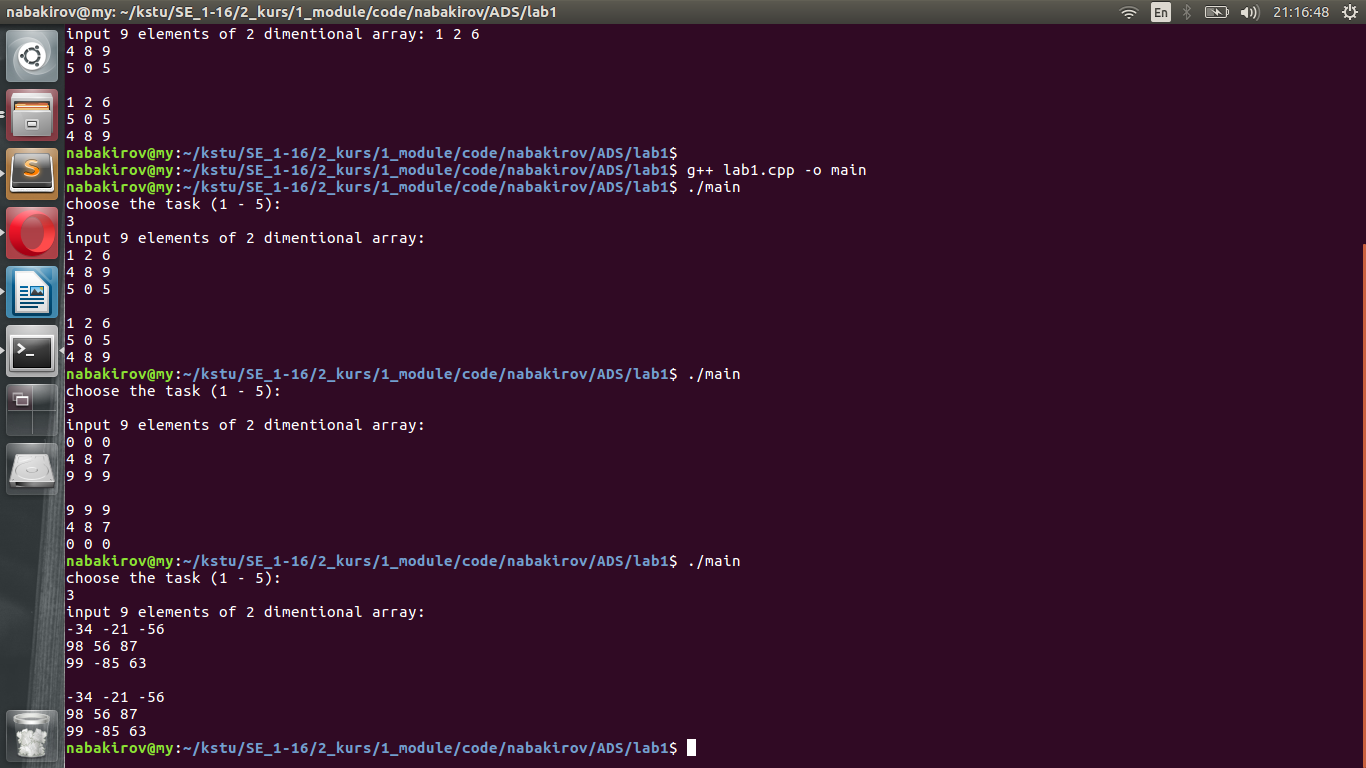
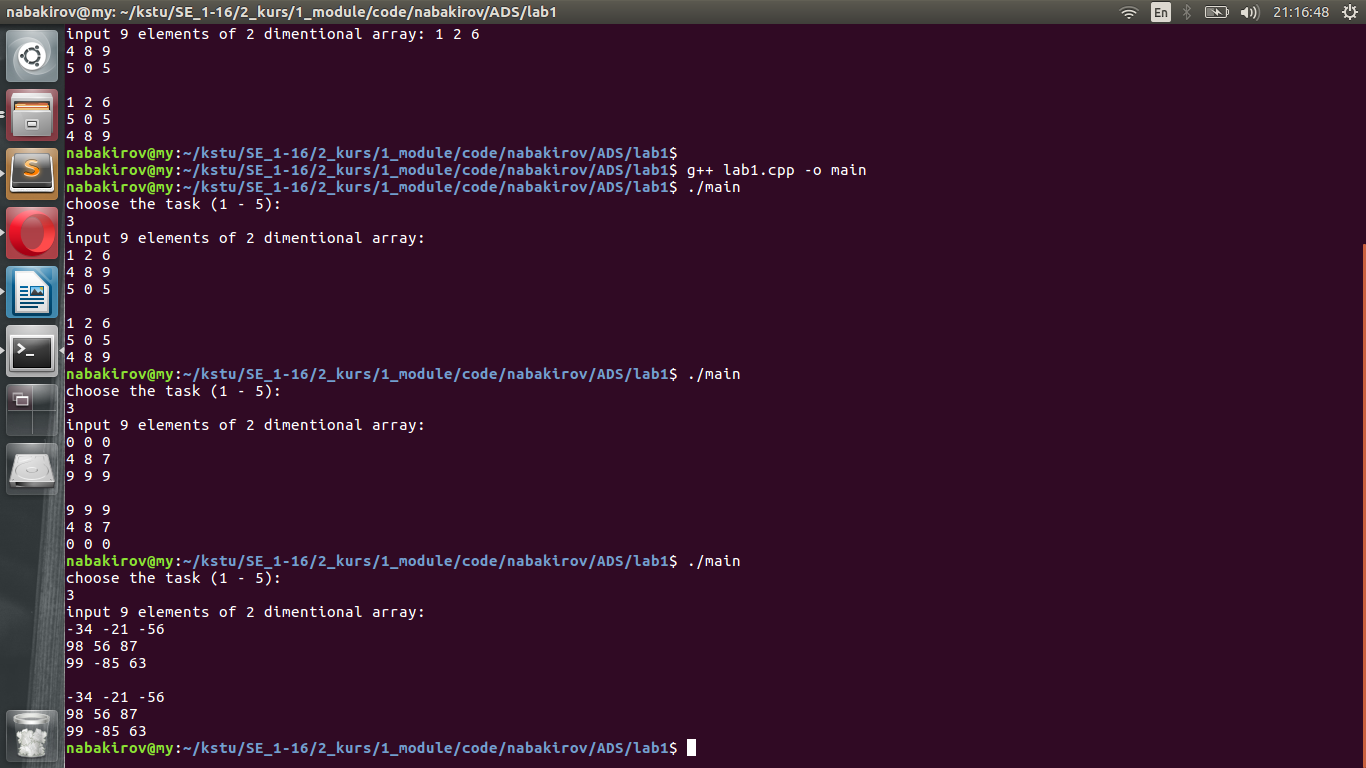
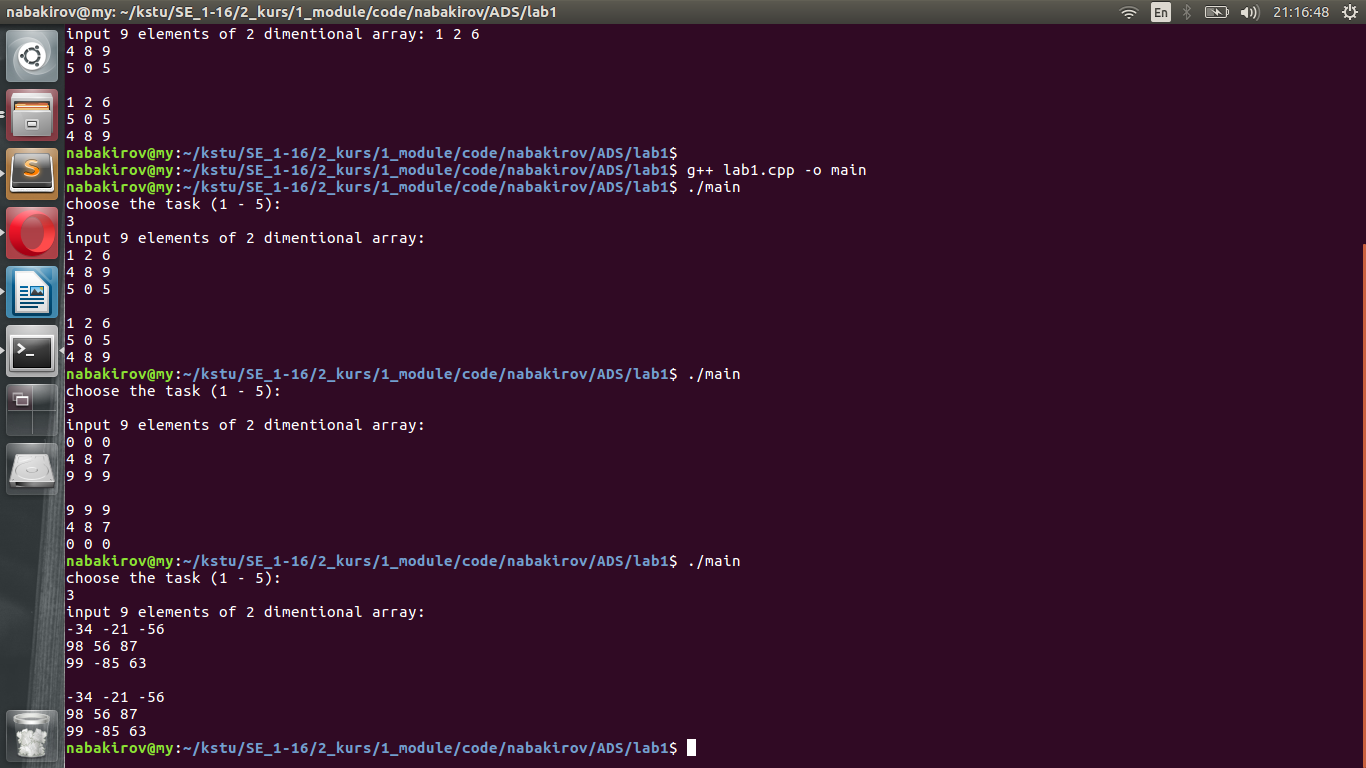
5 0 5 9 9 9 99 -85 63

Output: Output Output

1 2 6 9 9 9 -34 -21 -56

5 0 5 4 8 7 98 56 87

4 8 9 0 0 0 99 -85 63



Task 4

1. Problem statement:

Sort 1-dimensional array by ‘bubble’ method

2. Input and Output data

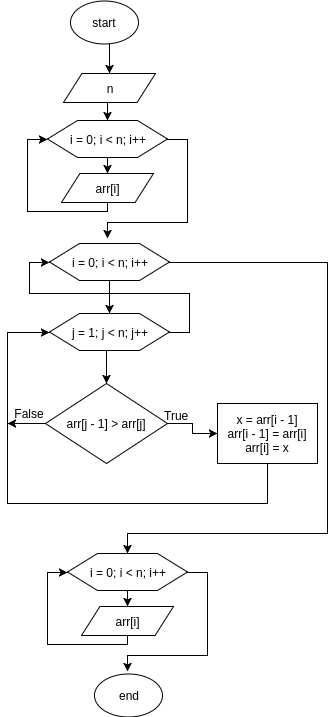
input data:

n – length of the array, integer

arr[n] – array of numbers, float or integer

output data:

arr[n] - array

3. Algorithm

#include <iostream>

using namespace std;

int main() {

short length;

cout << "input length of arrray: ";

cin >> length;

float \*arr = new float[length];

for (short i = 0; i < length; i++){

cin >> arr[i];

}

float x;

for(short j = 0; j < length; j++){

for (short i = 0; i < length; i++){

if (arr[i - 1] > arr[i]){

x = arr[i - 1];

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

arr[i] = x;

}

}

}

cout << endl;

for(short i = 0; i < length; i++){

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

}

delete [] arr;

}

Tests

1: 3:

input: Input:

5 4

0 1 2 3 4 -8 -9 5 -7

Output: Output:

0 1 2 3 4 -9 -8 -7 5

2: 4:

Input: Input:

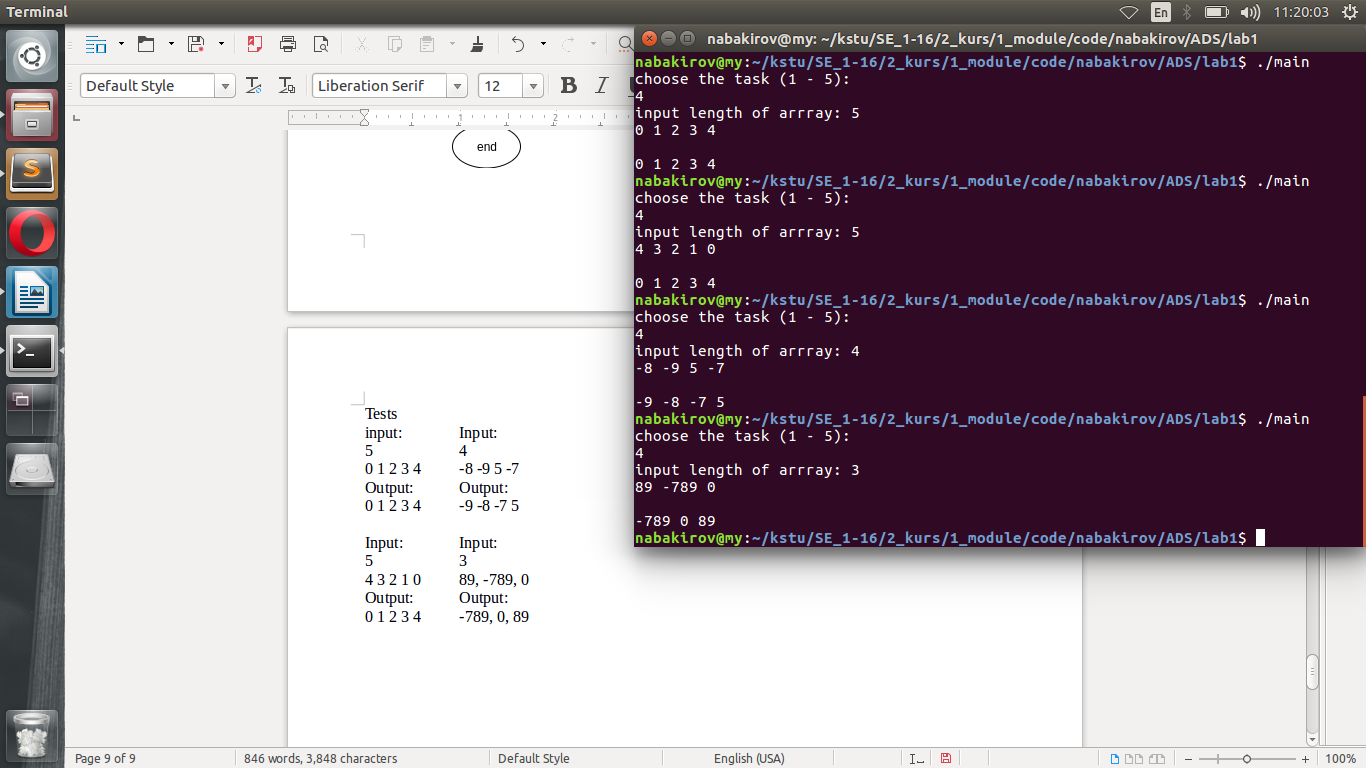
5 3

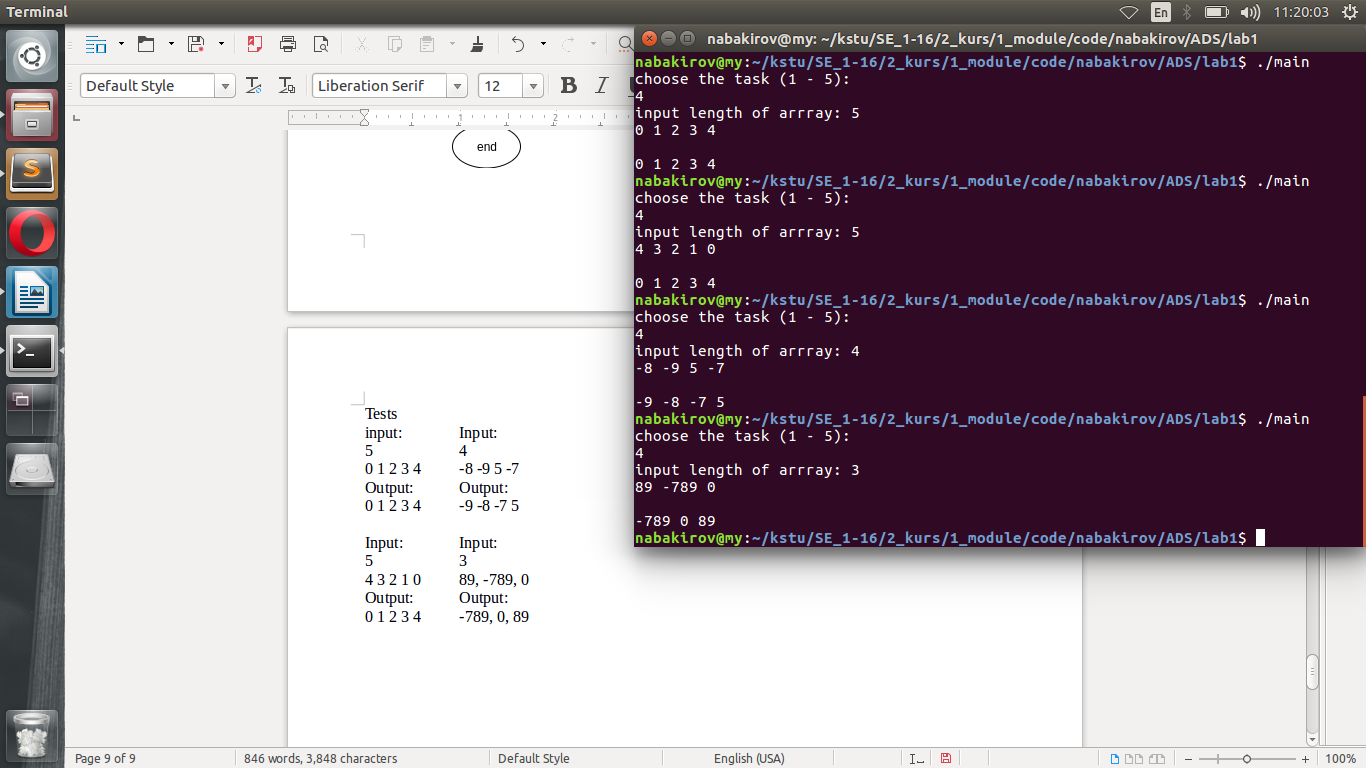
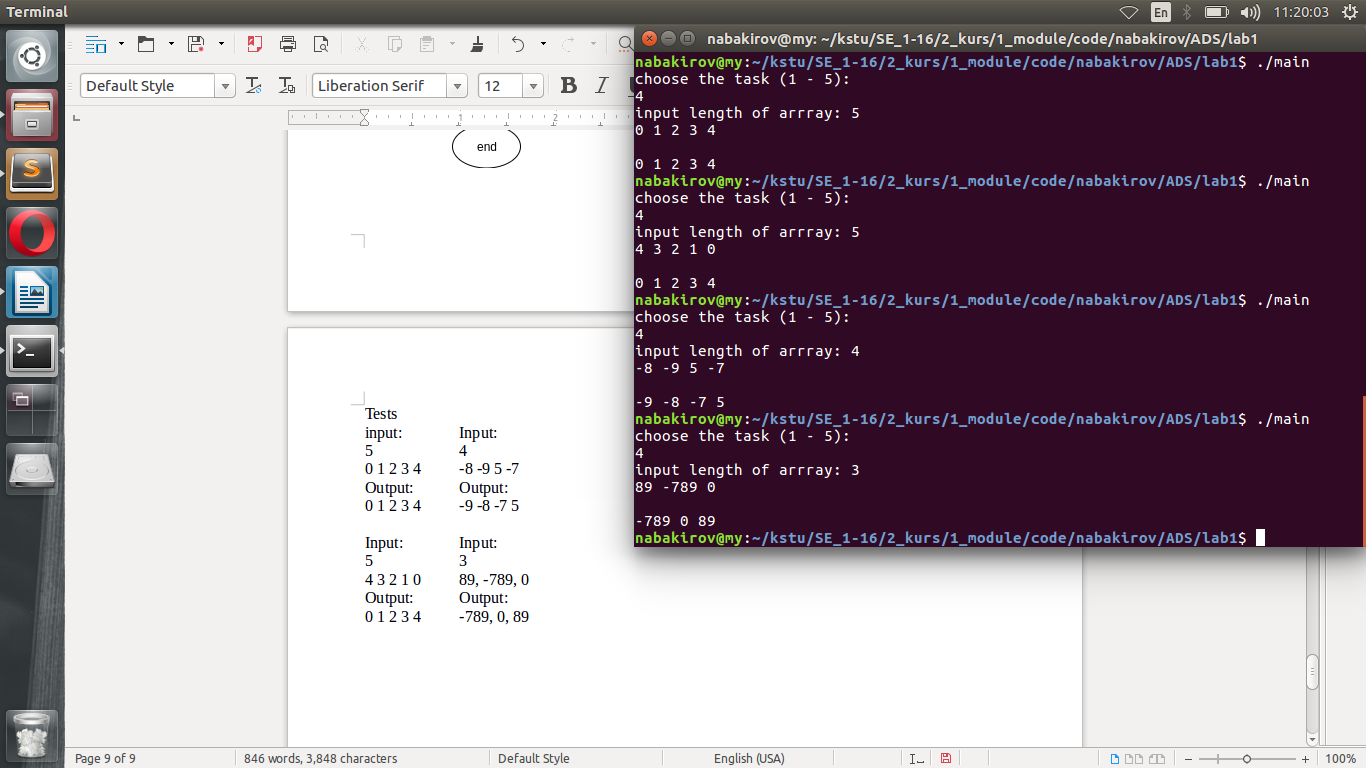
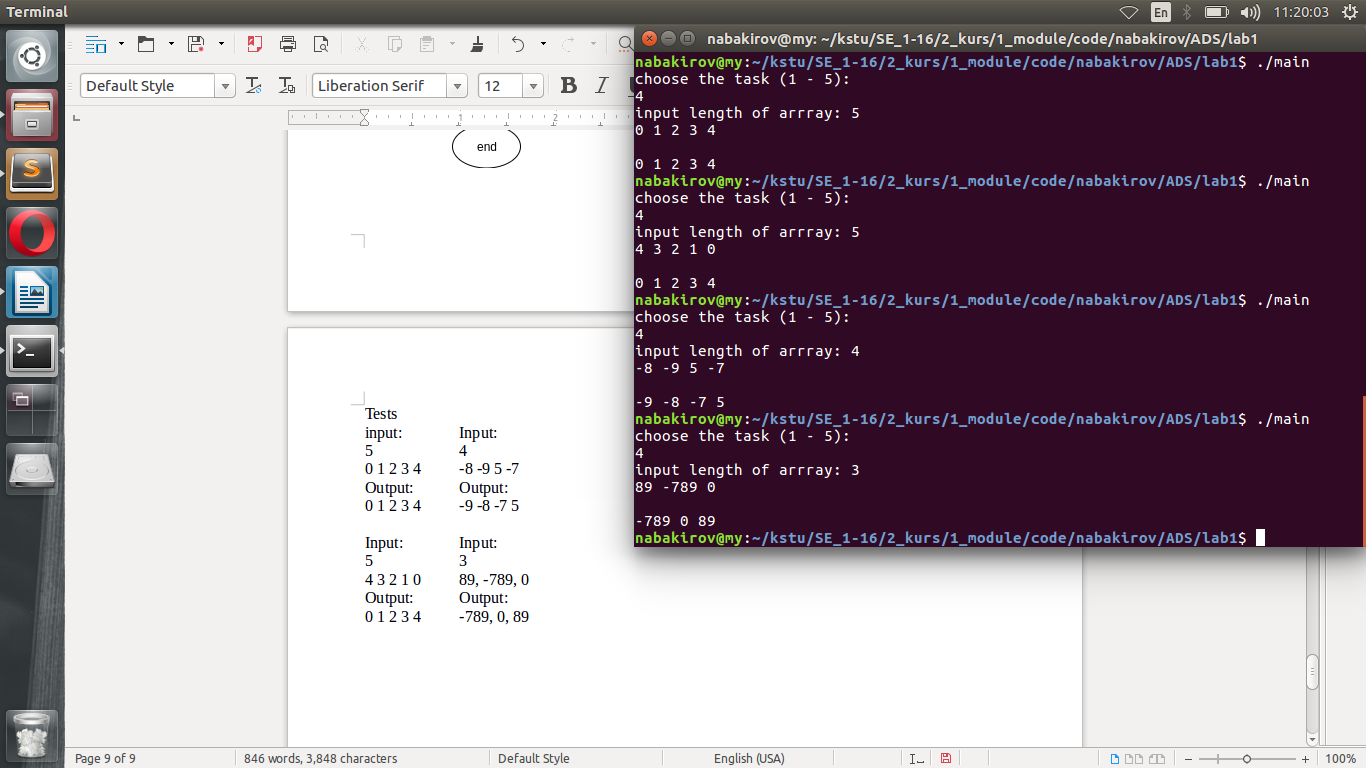
4 3 2 1 0 89, -789, 0

Output: Output:

0 1 2 3 4 -789, 0, 89

1:



2:

3:

4: