

ЗВІТ З ЛАБОРАТОРНОЇ РОБОТИ №7

за курсом “Програмування”

студентки групи ПА-23-1

Мазур Віолети

кафедра комп'ютерних технологій, ДНУ

2023/2024

1. Постановка задачі.

Завдання:

Складіть програму, яка здійснює обробку вхідних даних за допомогою одновимірних масивів. Для вводу даних і виводу результатів програма використовує текстові файли за допомогою перенаправлення у командному рядку, ввід з клавіатури також можливий. Кількість елементів масивів задається на початку файлу (вважаємо, що користувач всі дані вводить вірно). Програма повинна містити не менше 3 функцій. Одну із підзадач вирішити за допомогою вказівників при обробці масиву. Глобальні змінні не використовуємо. Розмірність одновимірних масивів при демонстрації виконання завдання повинна бути не меншою 15.

Варіант 11.

Задайте два масиви $A(n)$ і $B(m)$. Виконайте такі завдання:

- визначте масив $C(n+m)$ з елементів масивів A і B , спочатку парних у порядку спадання, а потім непарних у порядку зростання;
- знайдіть добуток елементів масиву A , розташованих між максимальними і мінімальними за модулем елементами;
- упорядкуйте елементи масивів за спаданням модулів елементів.

2. Опис розв'язку

Для спрощення коду, для виконання окремих дій в проєкті, створено функції:

1. `void printArray(int* (arr), int sizeArr);` - виведення значень елементів масиву в потік виведення. Функція перебирає елементи масиву і виводить їх в рядок. Після виведення функція додає перехід на новий рядок.

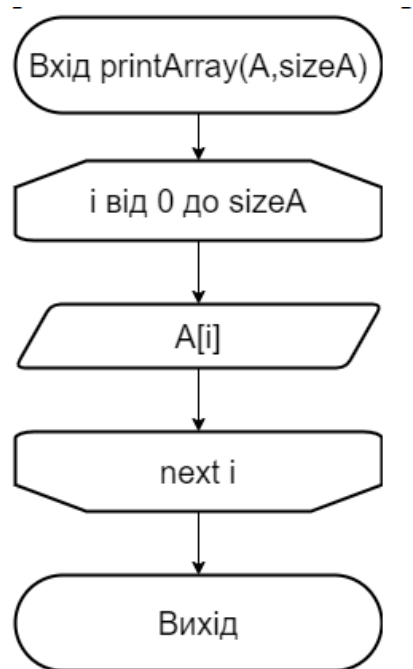


Рисунок 1. Блок-схема функції printArray()

2. void sortArray(int* (arr), int sizeArr, bool byGrowth); Функція сортує масив метом вибору. Аргумент byGrowth визначає правило сортування: значення true – сортування за зростанням, false – за спаданням.

За алгоритмом(рис.2):

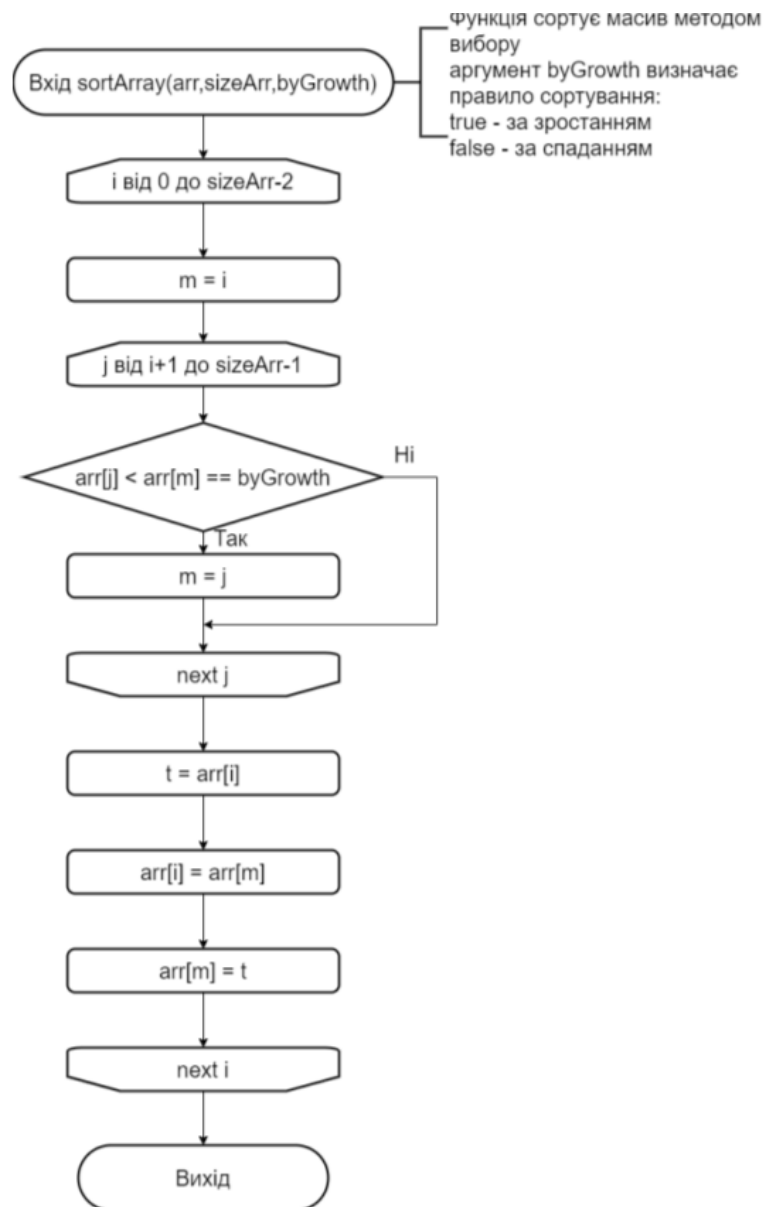


Рисунок 2. Блок-схема функції sortArray()

3. void sortArrayByAbs(int* (arr), int sizeArr); Функція сортує масив arr за спаданням абсолютної величини елемента.

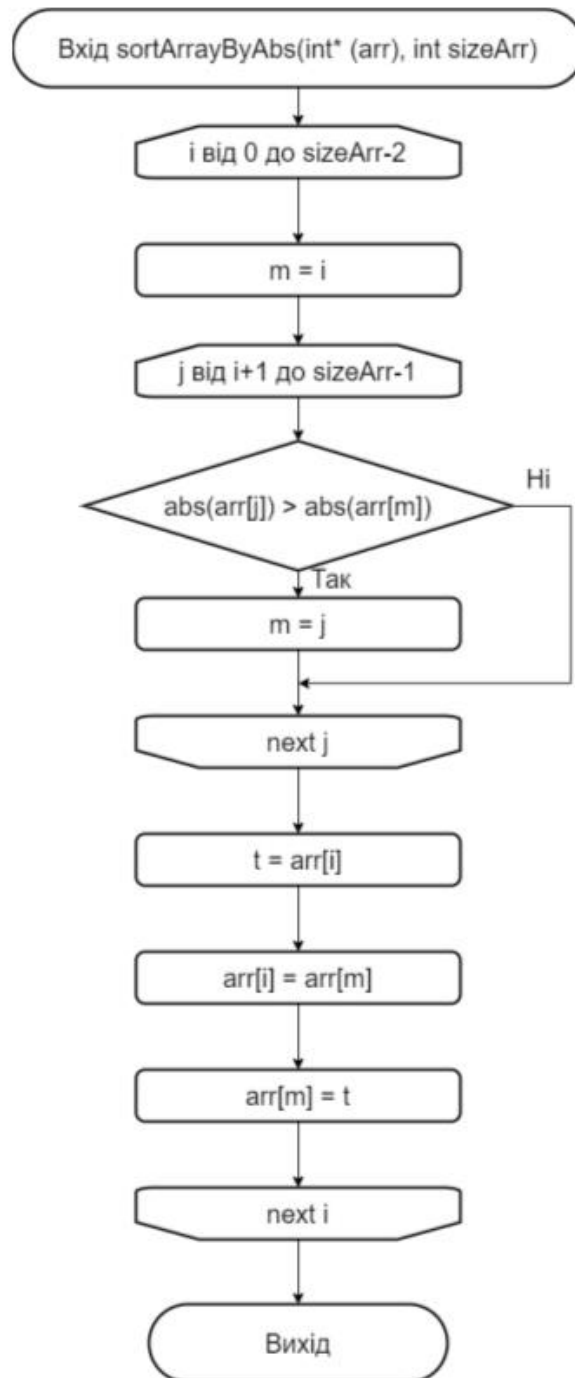


Рисунок 3. Блок-схема функції sortArrayByAbs()

4. void createArray(int *(A), int sizeA, int* (B), int sizeB, int* (C), int sizeC);

Функція створює масив C за правилами, спочатку парні елементи відсортовані за спаданням, потім непарні елементи відсортовані за зростанням.

Порядок дій:

1. Обчислення кількості парних елементів

- 2.Обчислення кількості непарних елементів
- 3.Створення масиву парних елементів
- 4.Створення масиву непарних елементів
- 5.Сортування масиву парних елементів за спаданням
- 6.Сортування масиву непарних елементів за зростанням
- 7.Злиття масивів

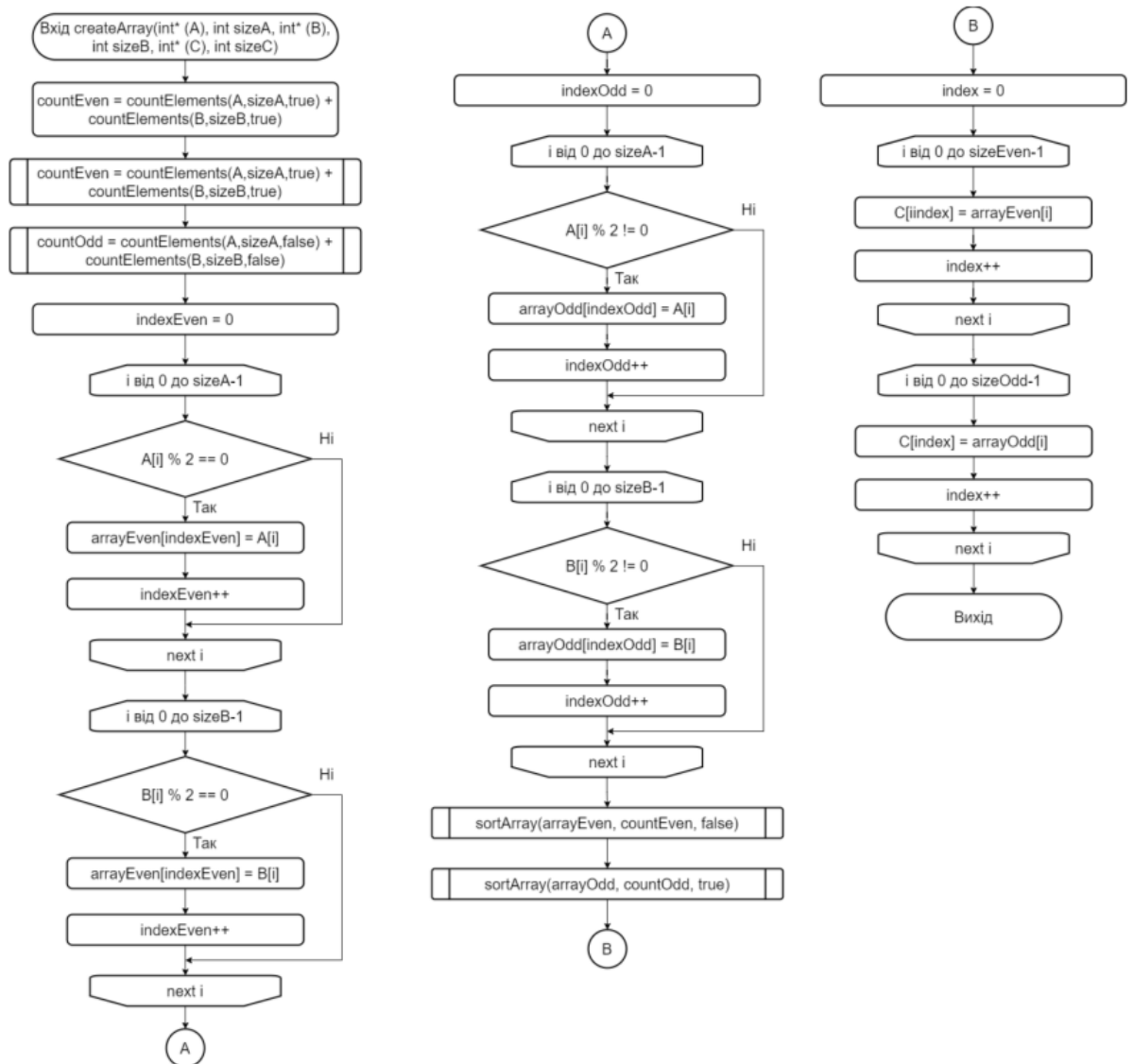


Рисунок 4. Блок-схема функції createArray()

5.int countElements(int* (A), int sizeA, bool even); Функція підраховує та повертає кількість парних (або непарних) елементів масиву. Парність визначається параметром even: true – парні, false – непарні.

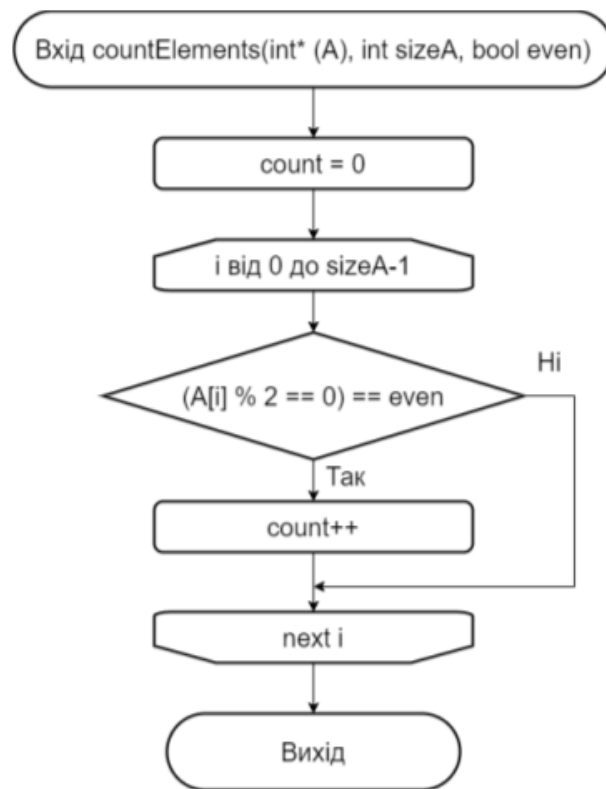


Рисунок 5. Блок-схема функції countElements()

6.int productElements(int* (A), int sizeA); Функція обчислює добуток елементів масиву, що знаходяться між найбільшим та найменшим за модулем елементами (включаючи найбільший та найменший)

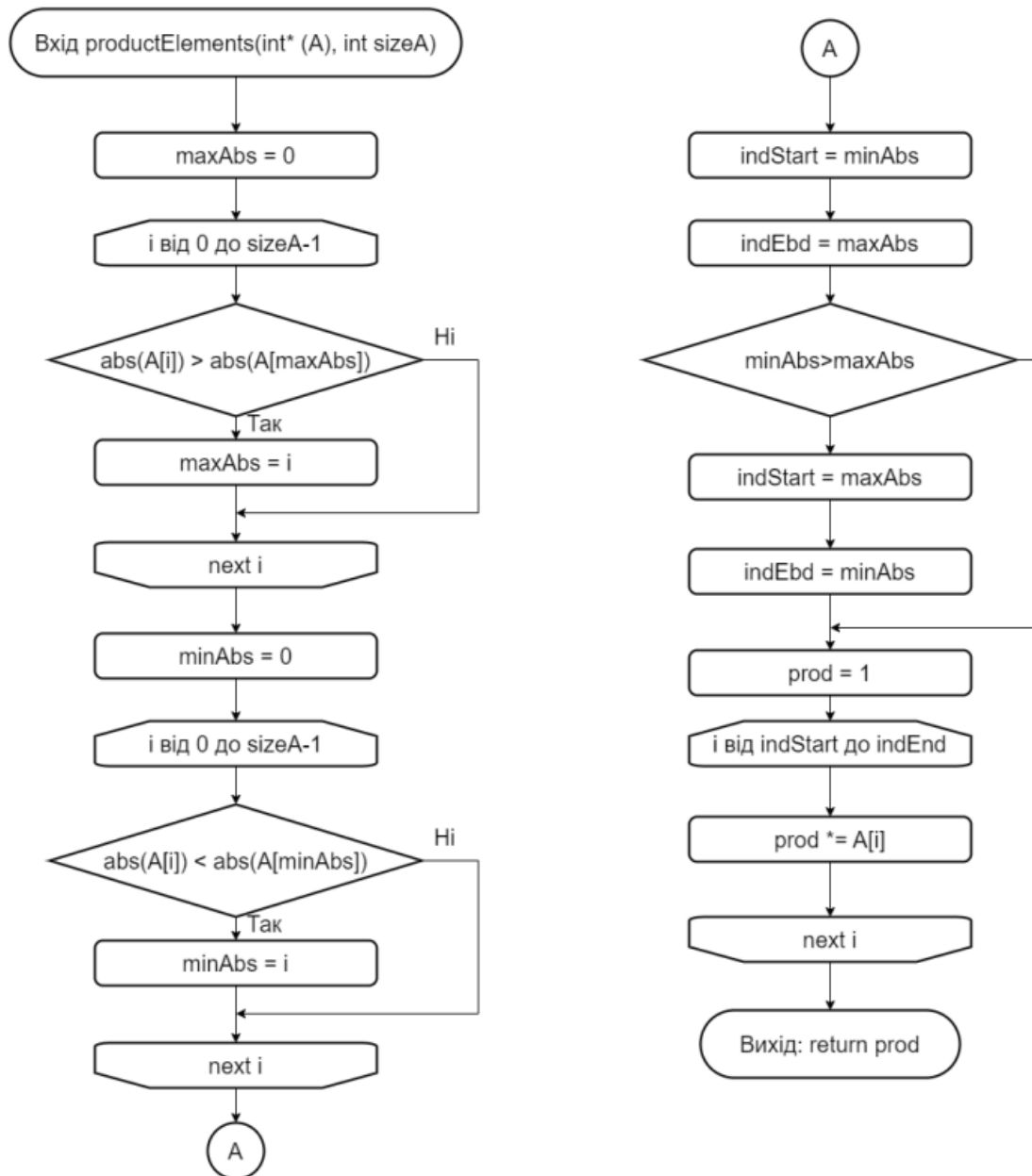


Рисунок 6. Блок-схема функції productElements()

7.Основна функція main.

Порядок дій:

- 1.Зчитування значення sizeA, sizeB
- 2.В циклі зчитування sizeA елементів в масив A
- 3.В циклі зчитування sizeB елементів в масив B
- 4.Визначення кількості елементів масиву C як суми sizeA+ sizeB
- 5.Створення і заповнення масиву C
- 6.Виведення масиву C

7.Визначення добутку елементів масиву А, що стоять між найменшим за модулем елементом і найбільшим за модулем елементом

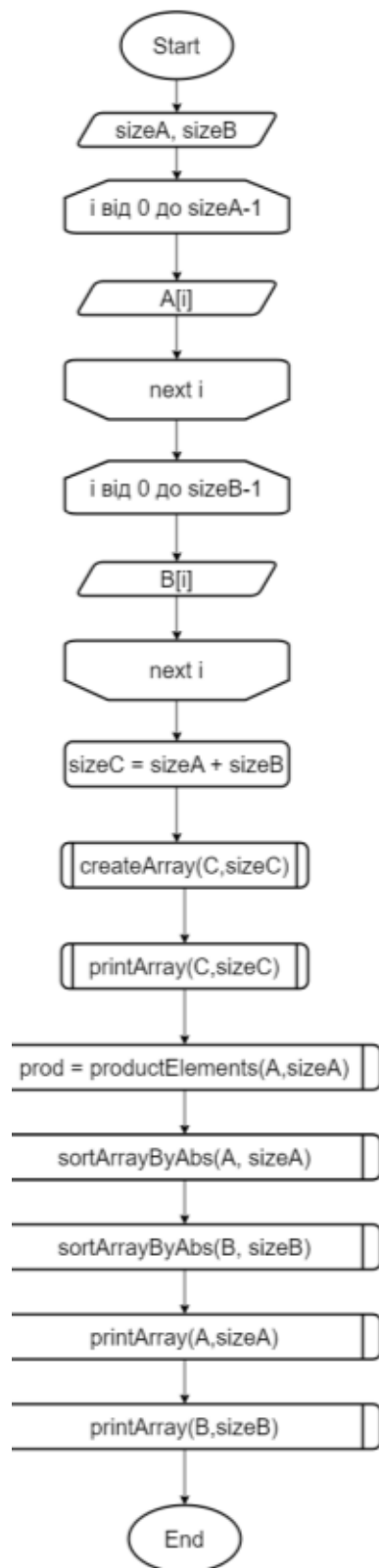
8.Виведення добутку

9.Сортування масиву А за модулем

10.Сортування масиву В за модулем

11.Виведення масиву А

12.Виведення масиву В



3. Вихідний текст програми

/*

* Варіант 11. Задайте два масиви A(n) і B(m). Виконайте такі завдання:

- * – визначте масив $C(n+m)$ з елементів масивів A і B , спочатку парних у порядку
- * спадання, а потім непарних у порядку зростання;
- * – знайдіть добуток елементів масиву A , розташованих між максимальними і
- * мінімальними за модулем елементами;
- * – упорядкуйте елементи масивів за спаданням модулів елементів
- */

```
#include <iostream>
#include <fstream>
using namespace std;
void printArray(int* (arr), int sizeArr);
// функція сортує масив A за правилом byGrowth (true - зростанням, false -
спаданням)
void sortArray(int* (arr), int sizeArr, bool byGrowth);
void sortArrayByAbs(int* (arr), int sizeArr);
void createArray(int *(A), int sizeA, int* (B), int sizeB, int* (C), int sizeC);
int countElements(int* (A), int sizeA, bool even);
int productElements(int* (A), int sizeA);
int main()
{

    // визначення вказівників

    int sizeA, sizeB, sizeC;

    // Читаю масиви з потоку
```

```

// Дані в потоці подано в вигляді 3 рядків.
// 1 - 2 числа, довжина масиву А та довжина масиву В
// 2 - елементи масиву А
// 3 - елементи масиву В

// читаю довжини масивів з першого рядка

cout << "Введіть розміри масивів А та В (2 числа) : \n";
cin >> sizeA >> sizeB;
// читаю перший масив
cout << "Введіть елементи масиву А," << sizeA << " чисел через Enter: \n";
int* A = new int[sizeA];
for (int i = 0; i < sizeA; i++) {
    cin >> A[i];
}
// читаю другий масив
cout << "Введіть елементи масиву В," << sizeB << " чисел через Enter: \n";
int* B = new int[sizeB];
for (int i = 0; i < sizeB; i++) {
    cin >> B[i];
}

cout << "1. Визначте масив C(n+m) з елементів масивів А і В, спочатку парних у порядку спадання, а потім непарних у порядку зростання; \n";
cout << "Масив C:\n";
sizeC = sizeA + sizeB;
int* C = new int[sizeC];
createArray(A,sizeA,B,sizeB,C,sizeC);
printArray(C,sizeC);

```

```

    cout << endl;

    cout << "2. знайдіть добуток елементів масиву A, розташованих між
    максимальними і мінімальними за модулем елементами\n";

    int prod = productElements(A,sizeA);
    cout << "Результат => " << prod << endl;
    cout << endl;

    cout << "3. упорядкуйте елементи масивів за спаданням модулів
    елементів \n";

    sortArrayByAbs(A, sizeA);
    sortArrayByAbs(B, sizeB);


    cout << "Масив A:\n";
    printArray(A, sizeA);
    cout << "Масив B:\n";
    printArray(B, sizeB);


    delete A;
    delete B;
    delete C;
    system("pause");
    return 0;
}

void printArray(int* (arr), int sizeArr) {
    for (int i = 0; i < sizeArr; i++) {
        cout << *(arr+i) << " ";
    }
}

```

```

    cout << endl;
}

void sortArray(int* (arr), int sizeArr, bool byGrowth) {
    // Сортування методом вибору
    for (int i = 0; i < sizeArr - 1; i++) {
        int m = i;
        for (int j = i+1; j < sizeArr; j++) {
            if (arr[j] < arr[m] == byGrowth) {
                m = j;
            }
        }
        int t = arr[i];
        arr[i] = arr[m];
        arr[m] = t;
    }
}

void sortArrayByAbs(int* (arr), int sizeArr) {
    // Сортування методом вибору
    for (int i = 0; i < sizeArr - 1; i++) {
        int m = i;
        for (int j = i + 1; j < sizeArr; j++) {
            if (abs(arr[j]) > abs(arr[m])) {
                m = j;
            }
        }
        int t = arr[i];

```

```

        arr[i] = arr[m];
        arr[m] = t;
    }
}

int countElements(int* (A), int sizeA, bool even) {
    int countEven = 0;
    for (int i = 0; i < sizeA; i++) {
        if ((abs(A[i]) % 2 == 0) == even) {
            countEven++;
        }
    }
    return countEven;
}

void createArray(int* (A), int sizeA, int* (B), int sizeB, int* (C), int sizeC) {
    /*
        * – визначте масив C(n+m) з елементів масивів A і B, спочатку парних у
        порядку
        * спадання, а потім непарних у порядку зростання;
        */
    // Довжина масиву C. В нього входять всі елементи A і B

    // Визначаю кількість парних та непарних елементів в масивах A та B
    int countEven = countElements(A, sizeA, true) + countElements(B, sizeB, true);
    int countOdd = countElements(A, sizeA, false) + countElements(B, sizeB, false);
    // Створюю масив для парних елементів
    int* arrayEven = new int[countEven];
    int indexEven = 0;
    for (int i = 0; i < sizeA; i++) {
        if (abs(A[i]) % 2 == 0) {

```

```

        arrayEven[indexEven] = A[i];
        indexEven++;
    }
}
for (int i = 0; i < sizeB; i++) {
    if (abs(B[i]) % 2 == 0) {
        arrayEven[indexEven] = B[i];
        indexEven++;
    }
}
// Створюю масив для непарних елементів
int* arrayOdd = new int[countOdd];
int indexOdd = 0;
for (int i = 0; i < sizeA; i++) {
    if (abs(A[i]) % 2 == 1) {
        arrayOdd[indexOdd] = A[i];
        indexOdd++;
    }
}
for (int i = 0; i < sizeB; i++) {
    if (abs(B[i]) % 2 == 1) {
        arrayOdd[indexOdd] = B[i];
        indexOdd++;
    }
}
// Сортую масив з парними елементами за спаданням
sortArray(arrayEven, countEven, false);
// Сортую масив з непарними елементами за зростанням

```

```
sortArray(arrayOdd, countOdd, true);  
// З'єднує масиви  
int index = 0;  
for (int i = 0; i < countEven; i++) {  
    C[index] = arrayEven[i];  
    index++;  
}  
for (int i = 0; i < countOdd; i++) {  
    C[index] = arrayOdd[i];  
    index++;  
}  
  
}
```

```
int productElements(int* (A), int sizeA) {  
    // максимальний за модулем елемент  
    int maxAbs = 0;  
    for (int i = 0; i < sizeA; i++) {  
        if (abs(A[maxAbs]) < abs(A[i])) {  
            maxAbs = i;  
        }  
    }  
    // мінімальний за модулем елемент  
    int minAbs = 0;  
    for (int i = 0; i < sizeA; i++) {  
        if (abs(A[minAbs]) > abs(A[i])) {  
            minAbs = i;  
        }  
    }  
}
```



```

    }
}

// визначення початку та кінця діапазону елементів
int indStart = maxAbs, indEnd = minAbs;
if (maxAbs > minAbs) {
    indStart = minAbs;
    indEnd = maxAbs;
}
// визначення добутку
int prod = 1;
for (int i = indStart; i <= indEnd; i++) {
    prod *= A[i];
}
return prod;
}

```

4. Опис інтерфейсу програми.

Запускаєте програму файлом .exe.

Програма чекає на введення двох чисел, що є довжиною масиву А та довжиною масиву В відповідно.

Далі програма чекає введення елементів масиву А

Далі програма чекає введення елементів масиву В.

Після введення значень програма виводить на екран масив С, що є результатом виконання першого завдання. Наступним рядком програма виводить добуток елементів.

В наступних двох рядках вводяться відсортовані за модулем значення елементів масиву А та масиву В відповідно.

Якщо потрібно передати значення масиву А та В через текстовий файл та отримати їх в іншому текстовому файлі, то користуємось командним рядком, input_file.txt та output_file.txt. В файл input_file.txt вписуємо

кількість елементів масиву A та B, а потім самі елементи цих масивів. Після відкриваємо командний рядок, копіюємо та вставляємо шлях до .exe файлу програми, після пишемо «>» та вставляємо шлях до файлу output_file.txt та тиснемо на Enter. Програма взяла дані з файлу input_file.txt та результат виконання програми з цими даними перенесла в текстовий файл output_file.txt.

5. Опис тестових прикладів

```
Введіть розміри масивів A та B (2 числа) :
3
5
Введіть елементи масиву A, 3 чисел через Enter:
45
7
0
Введіть елементи масиву B, 5 чисел через Enter:
34
6778
2
-10
9
1. Визначте масив  $\diamond(n+m)$  з елементів масивів A і B, спочатку парних у порядку спадання, а потім непарних у порядку зростання;
Масив C:
6778 34 2 0 -10 7 9 45

2. знайдіть добуток елементів масиву A, розташованих між максимальними і мінімальними за модулем елементами
Результат => 0

3. упорядкуйте елементи масивів за спаданням модулів елементів
Масив A:
45 7 0
Масив B:
6778 34 -10 9 2
sh: 1: pause: not found

...Program finished with exit code 0
Press ENTER to exit console.
```

```
Введіть розміри масивів A та B (2 числа) :
2 6
Введіть елементи масиву A, 2 чисел через Enter:
5
567
Введіть елементи масиву B, 6 чисел через Enter:
8 95 3 678 5 1
1. Визначте масив  $\diamond(n+m)$  з елементів масивів A і B, спочатку парних у порядку спадання, а потім непарних у порядку зростання;
Масив C:
678 8 1 3 5 5 95 567

2. знайдіть добуток елементів масиву A, розташованих між максимальними і мінімальними за модулем елементами
Результат => 2835

3. упорядкуйте елементи масивів за спаданням модулів елементів
Масив A:
567 5
Масив B:
678 95 8 5 3 1
sh: 1: pause: not found

...Program finished with exit code 0
Press ENTER to exit console.
```

```

Введіть розміри масивів А та В (2 числа) :
5
10
Введіть елементи масиву А, 5 чисел через Enter:
15
6
87
1
3
Введіть елементи масиву В, 10 чисел через Enter:
5
243
76
87
1
9
54
4
6
7
1. Визначте масив  $\diamond(n+m)$  з елементів масивів А і В, спочатку парних у порядку спадання, а потім непарних у порядку зростання;
 $\diamond$ масив С:
76 54 6 6 4 1 1 3 5 7 9 15 87 87 243

2. знайдіть добуток елементів масиву А, розташованих між максимальними і мінімальними за модулем елементами
Результат => 87

3. упорядкуйте елементи масивів за спаданням модулів елементів
Масив А:
87 15 6 3 1
Масив В:
243 87 76 54 9 7 6 5 4 1
sh: 1: pause: not found

...Program finished with exit code 0
Press ENTER to exit console.

```

```

Введіть розміри масивів А та В (2 числа) :
3
6
Введіть елементи масиву А, 3 чисел через Enter:
100
10
15
Введіть елементи масиву В, 6 чисел через Enter:
45
450
5
78
1
98
1. Визначте масив  $\diamond(n+m)$  з елементів масивів А і В, спочатку парних у порядку спадання, а потім непарних у порядку зростання;
 $\diamond$ масив С:
450 100 98 78 10 1 5 15 45

2. знайдіть добуток елементів масиву А, розташованих між максимальними і мінімальними за модулем елементами
Результат => 1000

3. упорядкуйте елементи масивів за спаданням модулів елементів
Масив А:
100 15 10
Масив В:
450 98 78 45 5 1
sh: 1: pause: not found

...Program finished with exit code 0
Press ENTER to exit console.

```

```
Командний рядок
Microsoft Windows [Version 10.0.19045.4170]
(c) Корпорація Майкрософт. Усі права захищені.

C:\Users\Ceprii>d:

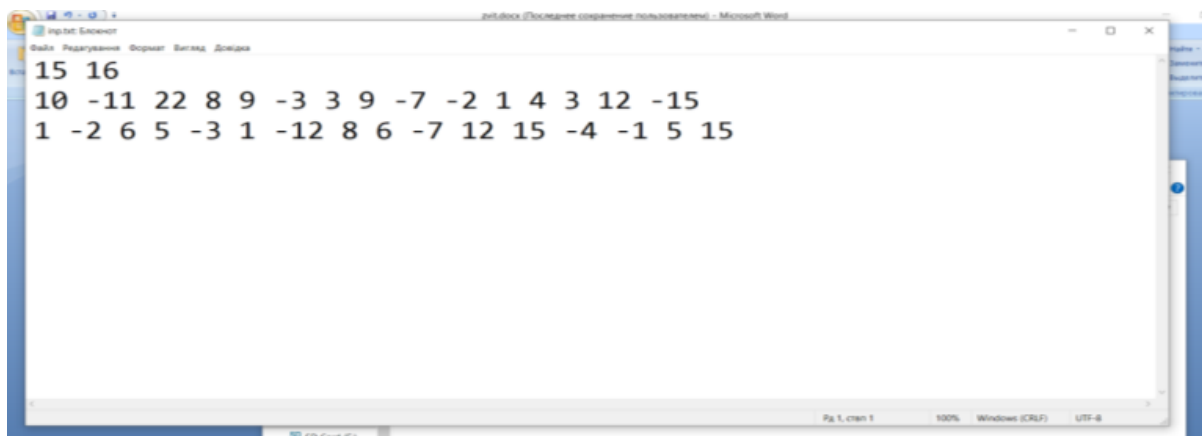
D:\>cd D:\ConsoleApplication1\x64\Debug

D:\ConsoleApplication1\x64\Debug>dinamicarray < inp.txt > out.txt
'dinamicarray' is not recognized as an internal or external command,
operable program or batch file.

D:\ConsoleApplication1\x64\Debug>dinamicarray < inp.txt > out.txt

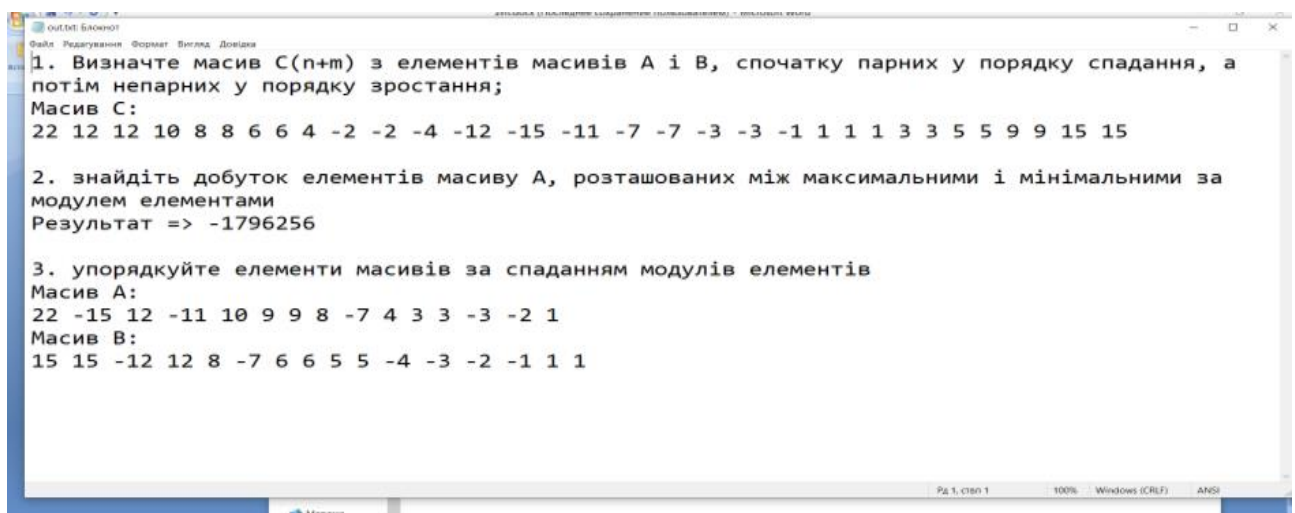
D:\ConsoleApplication1\x64\Debug>dinamicarray < inp.txt > out.txt
```

Файл з вхідними даними :



```
15 16
10 -11 22 8 9 -3 3 9 -7 -2 1 4 3 12 -15
1 -2 6 5 -3 1 -12 8 6 -7 12 15 -4 -1 5 15
```

Файл з вихідними даними :



```
1. Визначте масив C(n+m) з елементів масивів A і B, спочатку парних у порядку спадання, а
потім непарних у порядку зростання;
Масив C:
22 12 12 10 8 8 6 6 4 -2 -2 -4 -12 -15 -11 -7 -7 -3 -3 -1 1 1 1 3 3 5 5 9 9 15 15

2. знайдіть добуток елементів масиву A, розташованих між максимальними і мінімальними за
модулем елементами
Результат => -1796256

3. упорядкуйте елементи масивів за спаданням модулів елементів
Масив A:
22 -15 12 -11 10 9 9 8 -7 4 3 3 -3 -2 1
Масив B:
15 15 -12 12 8 -7 6 6 5 5 -4 -3 -2 -1 1 1
```

6. Аналіз помилок (опис усунення зауважень)

Під час розробки програми виникло декілька помилок, які були усунені. Була помилка у логіці сортування, яка призводила до неправильного порядку елементів у вихідному масиві. Я виправила умову порівняння у

функції сортування для забезпечення правильного порядку елементів. Також спочатку були проблеми з читанням даних з файлу та записом результатів у файл. Перевірила правильність шляхів до файлів та виправила формат введення/виведення даних.

7. Висновки

У цій лабораторній роботі було поставлено завдання створити два масиви $A(n)$ і $B(m)$ і виконати над ними різні операції. По-перше, визначила масив $C(n+m)$, об'єднавши елементи з масивів A і B . Елементи в масиві C були розташовані на дві частини – спочатку парні елементи в порядку спадання, потім непарні елементи в порядку зростання. Далі обчислила добуток елементів масиву A , які були розташовані між максимальним і мінімальним елементами за абсолютною величиною. Це передбачало визначення максимального та мінімального елементів у масиві A , обчислення їхніх абсолютних значень, а потім знаходження добутку елементів, розташованих між цими двома крайніми значеннями. Відсортовано елементи масивів у порядку спадання на основі абсолютних значень елементів. Цей процес вимагав обчислити абсолютні значення кожного елемента, відсортувати їх відповідно, а потім змінити порядок елементів у масивах A і B на основі цього критерію сортування. Загалом, поставлені завдання допомогли більш точно навчитись аналізувати масиви за допомогою різних математичних операцій і методів сортування, покращуючи розуміння маніпулювання масивами та алгоритмічного вирішення проблем.