Міністерство освіти і науки України

Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи №8 з дисципліни

«Алгоритми структури даних»

«Дослідження алгоритмів пошуку та сортування»

Варіант 34

Виконав студент ІП-1134 Шамков Іван Дмитрович
(прізвище, ім'я, по батькові)

Перевірив викладач Мартинова Оксана Петрівна
(прізвище, ім'я, по батькові)

Київ 2021

Лабораторна робота №8

Дослідження алгоритмів пошуку та сортування

Лабораторна робота 8

Дослідження алгоритмів пошуку та сортування

Мета – дослідити алгоритми пошуку та сортування, набути практичних навичок використання цих алгоритмів під час складання програмних специфікацій.

Варіант: 34

Умова задачі:

Завдання

Розробити алгоритм та написати програму, яка складається з наступних дій:

1. Опису змінної індексованого типу (двовимірний масив) згідно з варіантом (табл. 1).
2. Ініціювання змінної, що описана в п.1 даного завдання.
3. Створення нової змінної індексованого типу (одновимірний масив) та її ініціювання значеннями, що обчислюються згідно з варіантом (табл. 1).

34	5 x 7	Дійсний	Із середнього арифметичного додатних значень елементів стовпців двовимірного масиву. Відсортувати обміном за спаданням.

Математична модель:

Змінна	Тип	Ім’я	Призначення
Двовимірний масив	Дійсний	A	Початкове дане
Двовимірний масив	Дійсний	C	Результат
Лічильник	Цілий	i	Проміжне значення
Лічильник	Цілий	j	Проміжне значення
Сума додатних елементів стовпчика	Дійсний	sum	Проміжне значення
Кількість додатних елементів стовпчика	Цілий	count	Проміжне значення

Тимчасове значення, в яке записуємо A[i+1]	Дійсний	tmp	Проміжне значення

Постановка задачі:

Отже, математичне формулювання нашої задачі полягає в тому, щоб створити двовимірний масив, який наповнюємо випадковими дійсними числами. Після цього пробігаємося по стовпчикам цього масиву, обраховуючи середнє арифметичне додатних елементів кожного. При закінченні кожного стовпчику записуємо обраховане значення в другий одновимірний масив. Після заповнення цього масиву, сортуємо методом бульбашки(обміном) за спаданням.

(rand()%X - B) – генерація випадкового числа від з діапазону

[-B; X-B]

Наступні функції є створеними власноруч:

CreateArrayOneDimensional();

CreateArrayTwoDimensional();

CoutArrayTwoDimensional();

CoutArrayOneDimensional();

SortBubble();

Псевдокод:

Крок 1. Визначимо основні дії.

Крок 2. Деталізуємо значення A, C

Крок 3. Виведення A, C

Крок 4. Сортування C

Крок 5. Виведення C

Крок 1:

Start

Деталізуємо значення A, C

Виведення A, C

Сортування C

Виведення C

End

Крок 2:

Start

A=CreateArrayTwoDimensional()

C=CreateArrayOneDimensional(A)

Виведення A, C

Сортування C

Виведення C

End

Крок 3:

Start

A=CreateArrayTwoDimensional()

C=CreateArrayOneDimensional(A)

CoutArrayTwoDimensional(A)

CoutArrayOneDimensional(C)

Сортування C

Виведення C

End

Крок 4:

Start

A=CreateArrayTwoDimensional()

C=CreateArrayOneDimensional(A)

CoutArrayTwoDimensional(A)

CoutArrayOneDimensional(C)

C=SortBubble(C)

Виведення C

End

Крок 5:

Start

A=CreateArrayTwoDimensional()

C=CreateArrayOneDimensional(A)

CoutArrayTwoDimensional(A)

CoutArrayOneDimensional(C)

C=SortBubble(C)

CoutArrayOneDimensional(C)

End

Підпрограми

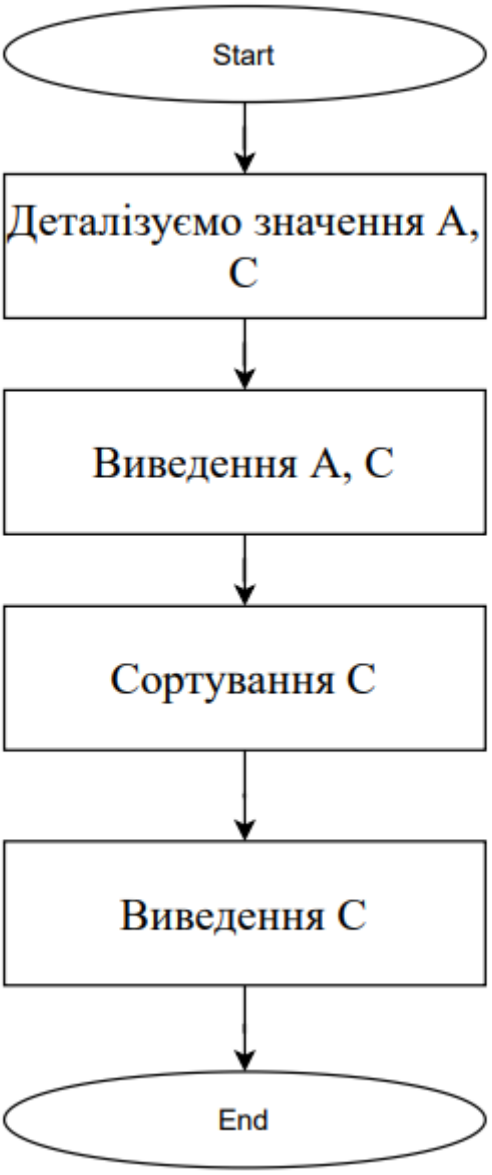
```
CreateArrayTwoDimensional(A[5][7])
        for j from 0 to 4
           repeat
                for i from 0 to 6
                   repeat
                        A[j][i] = (rand() % 2000 - 1000) / 10.0
                end for
        end for
        return A
CreateArrayOneDimensional(A[5][7])
   for i from 0 to 6
      repeat
        count=0
        sum=0
        for j from 0 to 4
           repeat
                if A[i][j]>0
                        sum+=A[j][i]
                        count+=1
                end if
        end for
        if count!=0
                C[i]=sum/count
        else
                C[i]=0
        end if
   end for
```

```
    return C


CoutArrayTwoDimensional (A[5][7])
        for j from 0 to 4
           repeat
                for i from 0 to 6
                   repeat
                        output A[j][i]
                end for
        end for
        return


CoutArrayOneDimensional(C)
        for i from 0 to 6
           repeat
                output C[i]
        end for
        return
```

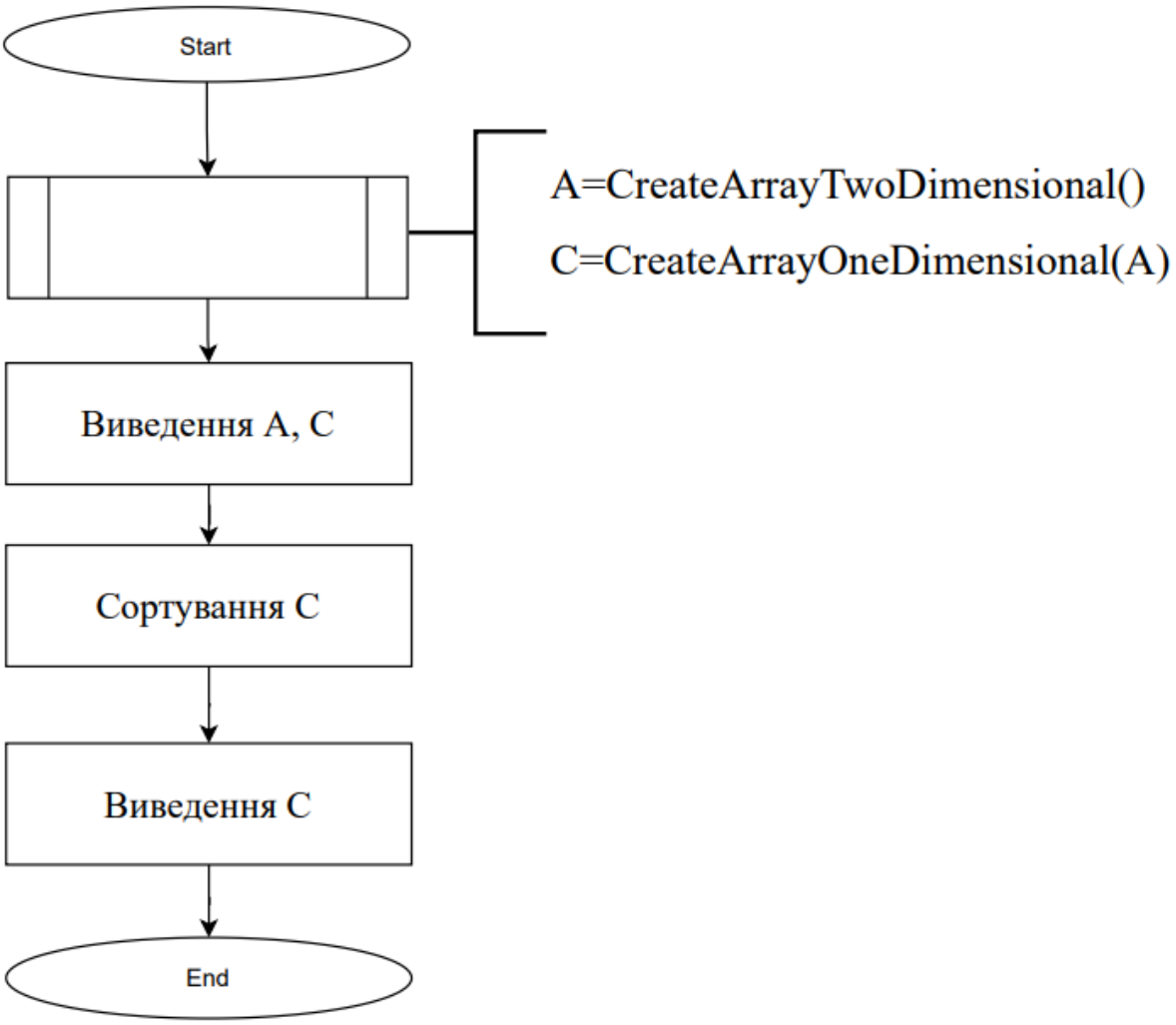
```
SortBubble(C)
        for i from 0 to 6
           repeat
                for j from 0 to 5
                   repeat
                        if (C[j]<C[j+1])
                                tmp = C[j+1]
                                C[j+1] = C[j]
                                C[j] = tmp
                        end if
                end for
        end for
        return
```

Блок схема:

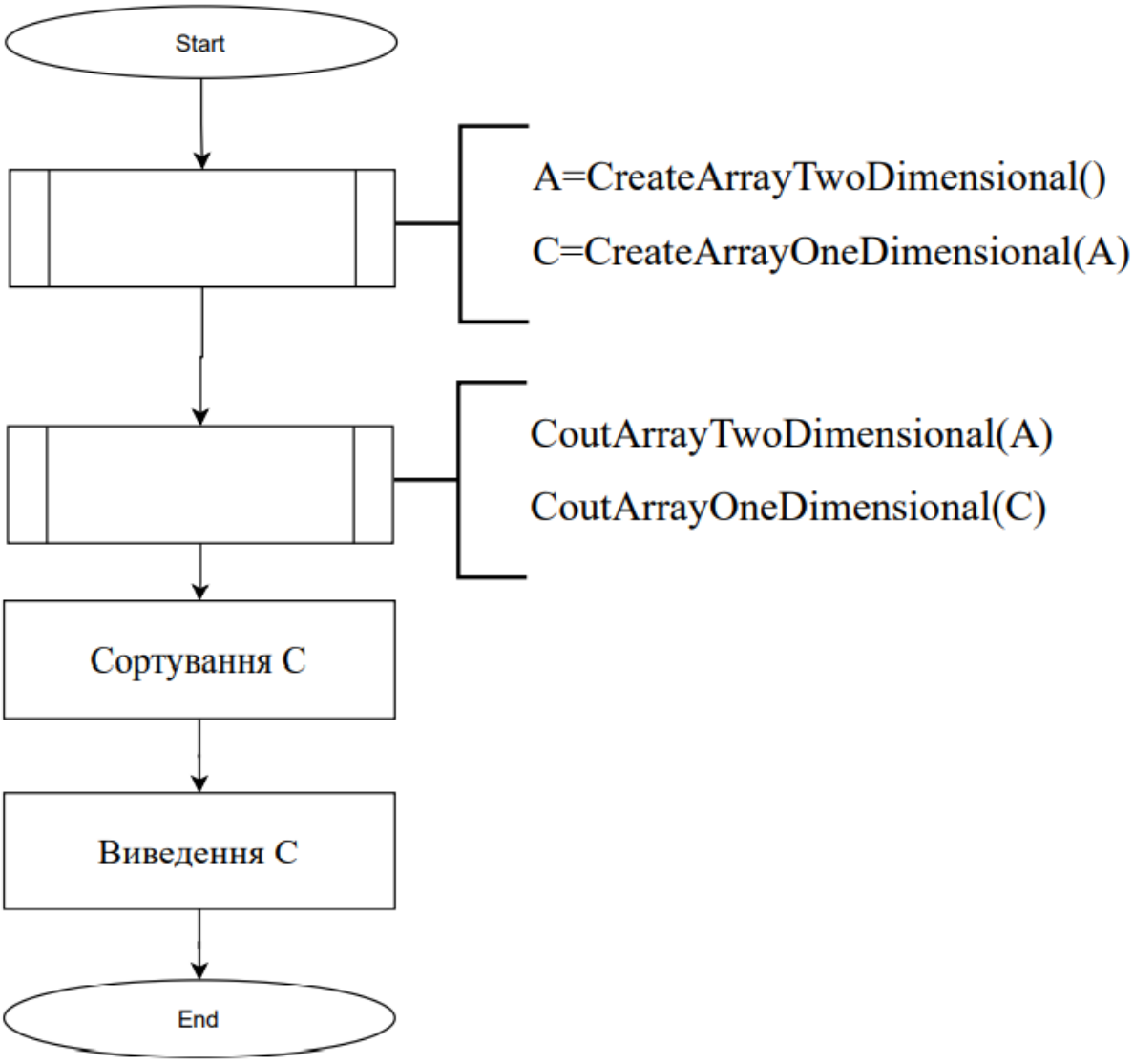
Крок 1



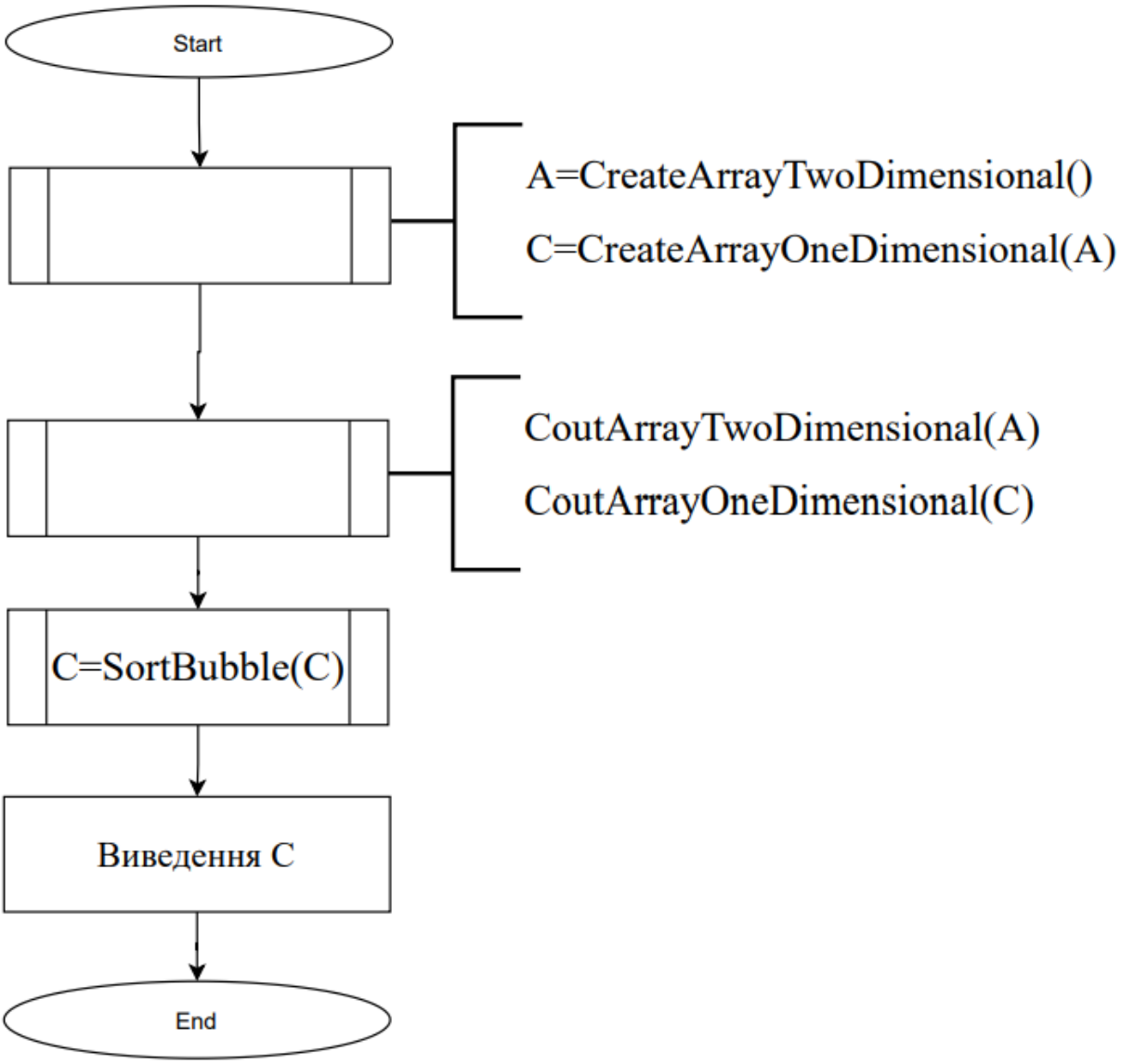
Крок 2


Start
A=CreateArrayTwoDimensional()
C=CreateArrayOneDimensional(A)
Виведення A, C
Сортування C
Виведення C
End

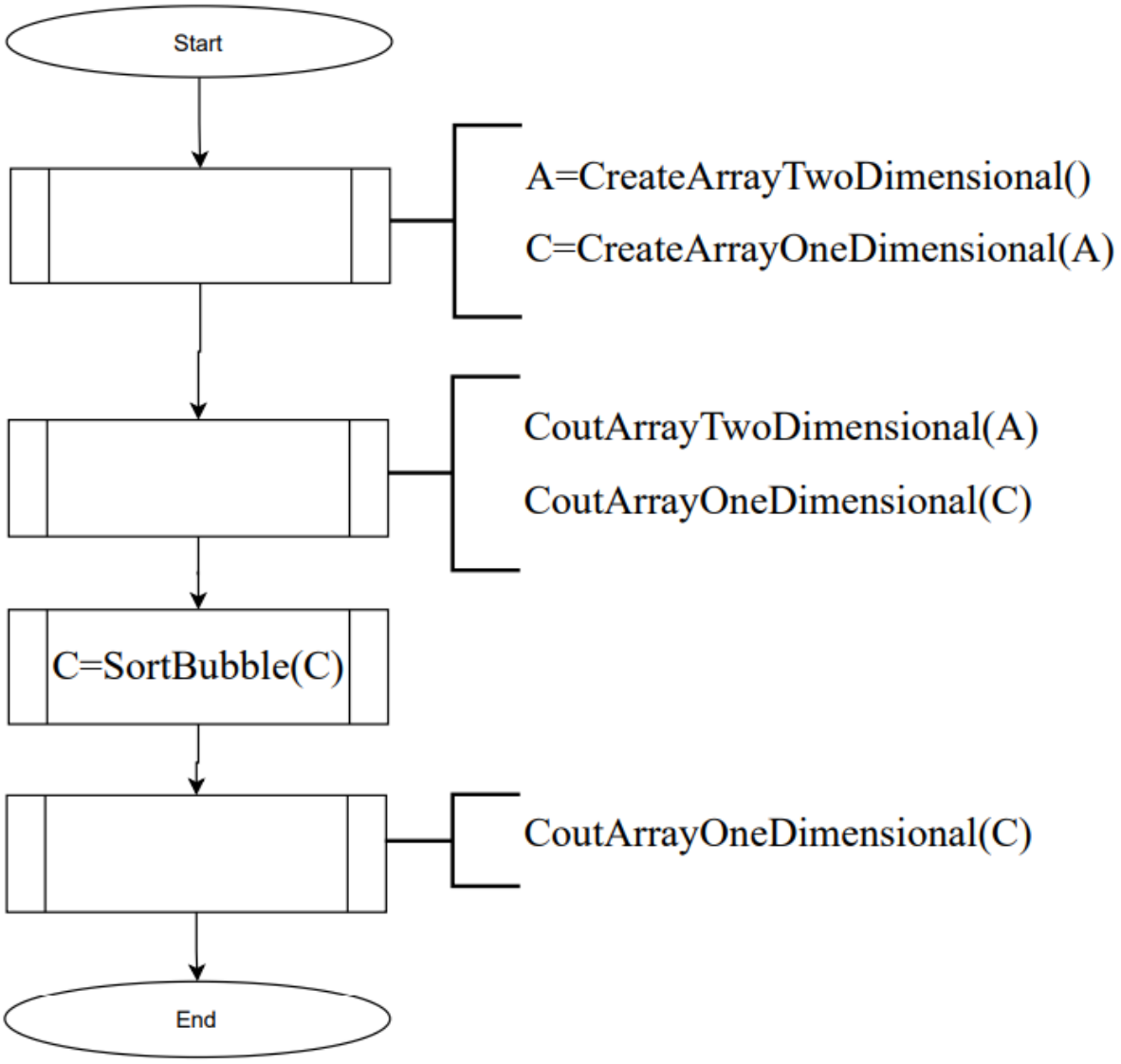
Крок 3


Start
A=CreateArrayTwoDimensional()
C=CreateArrayOneDimensional(A)
CoutArrayTwoDimensional(A)
CoutArrayOneDimensional(C)
Сортування С
Виведення С
End

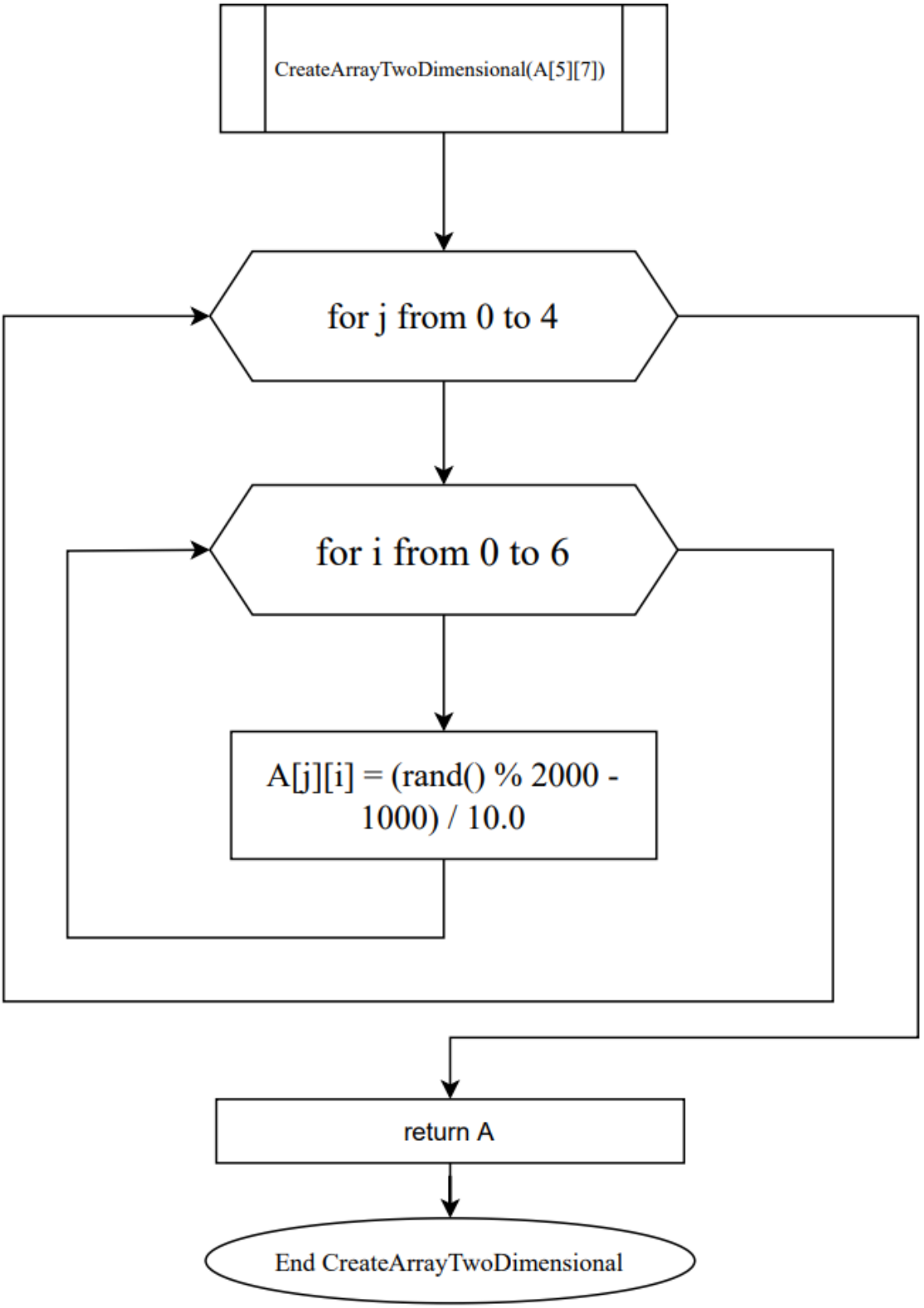
Крок 4

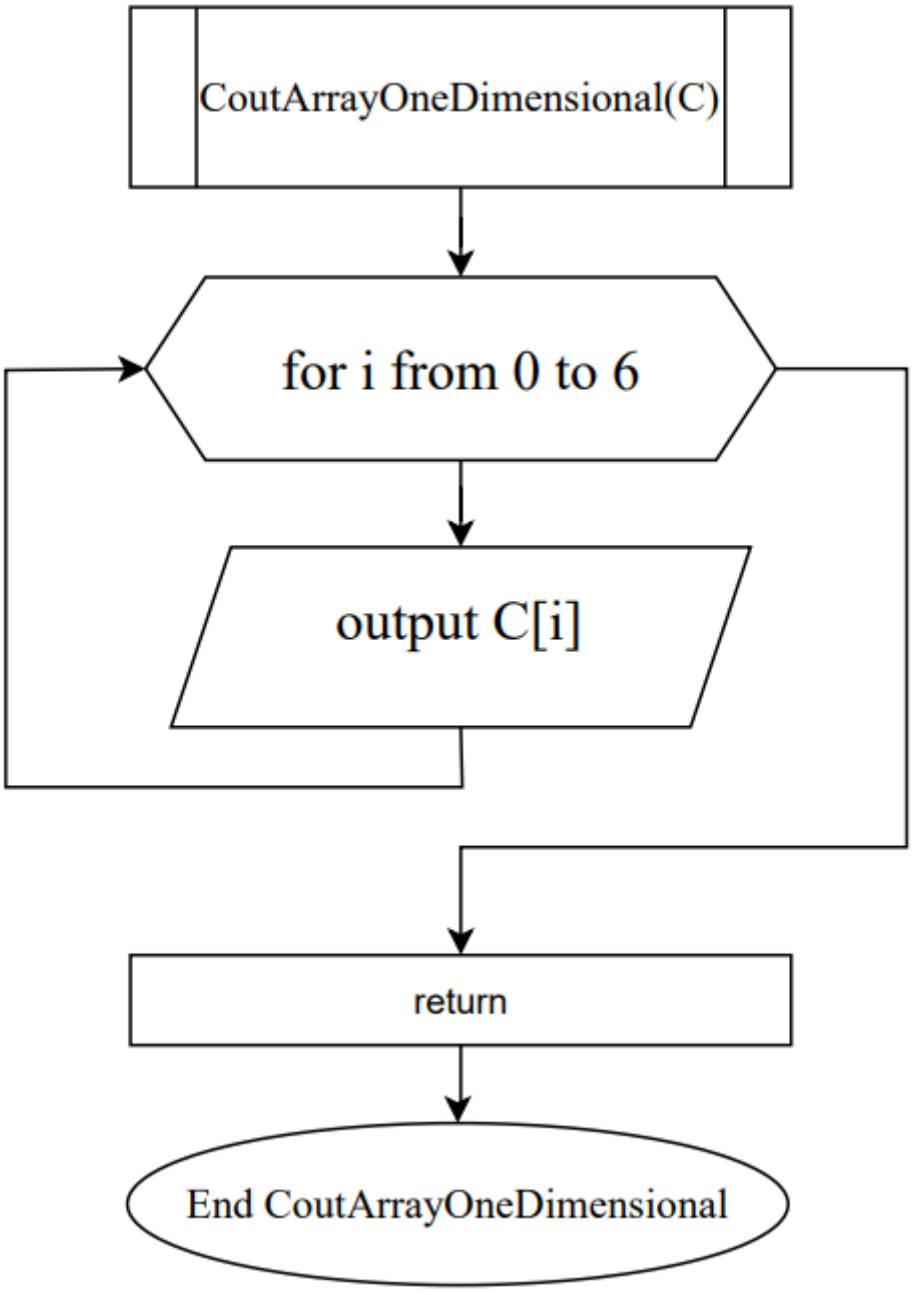

Start
A=CreateArrayTwoDimensional()
C=CreateArrayOneDimensional(A)
CoutArrayTwoDimensional(A)
CoutArrayOneDimensional(C)
C=SortBubble(C)
Виведення С
End

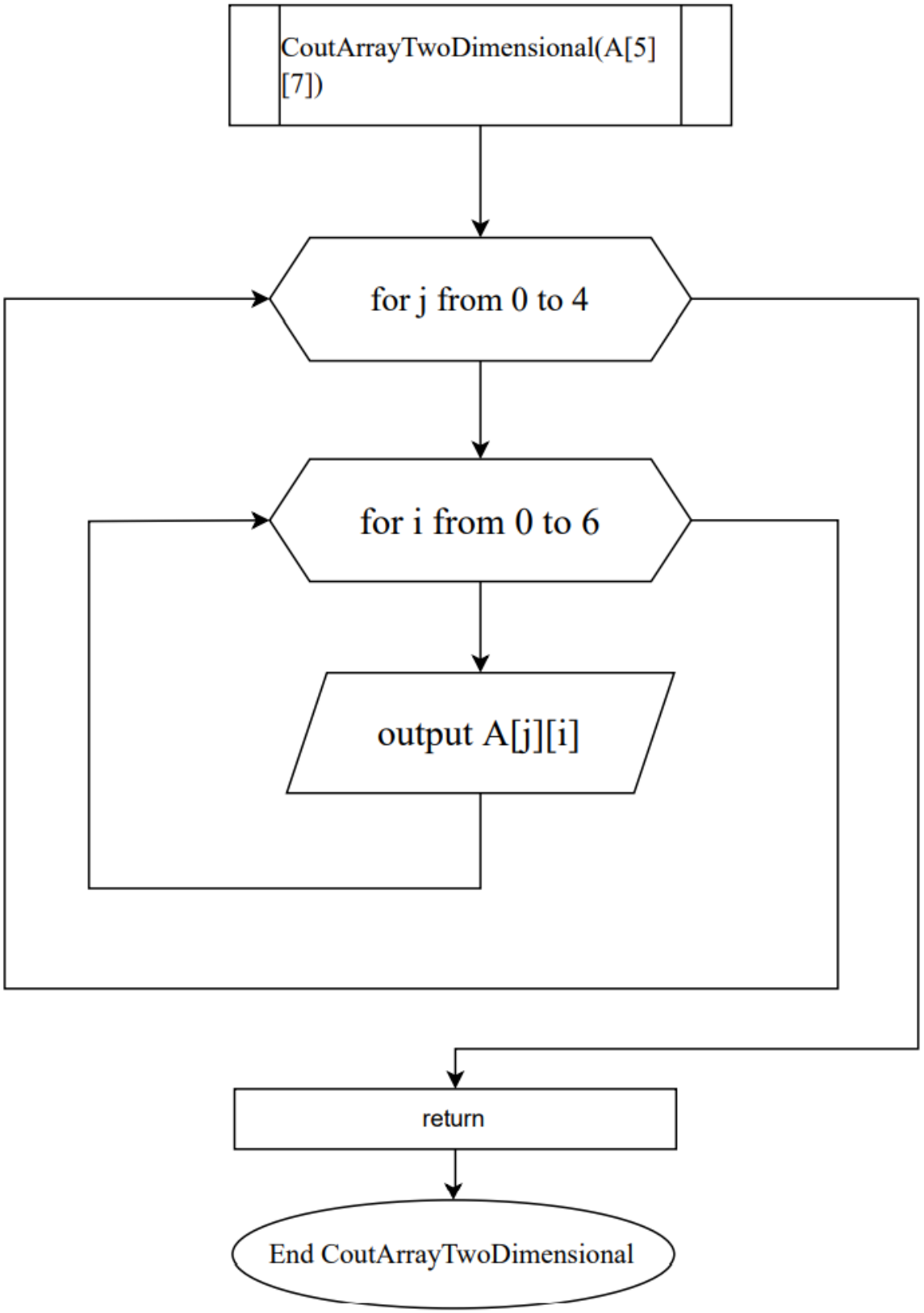
Крок 5

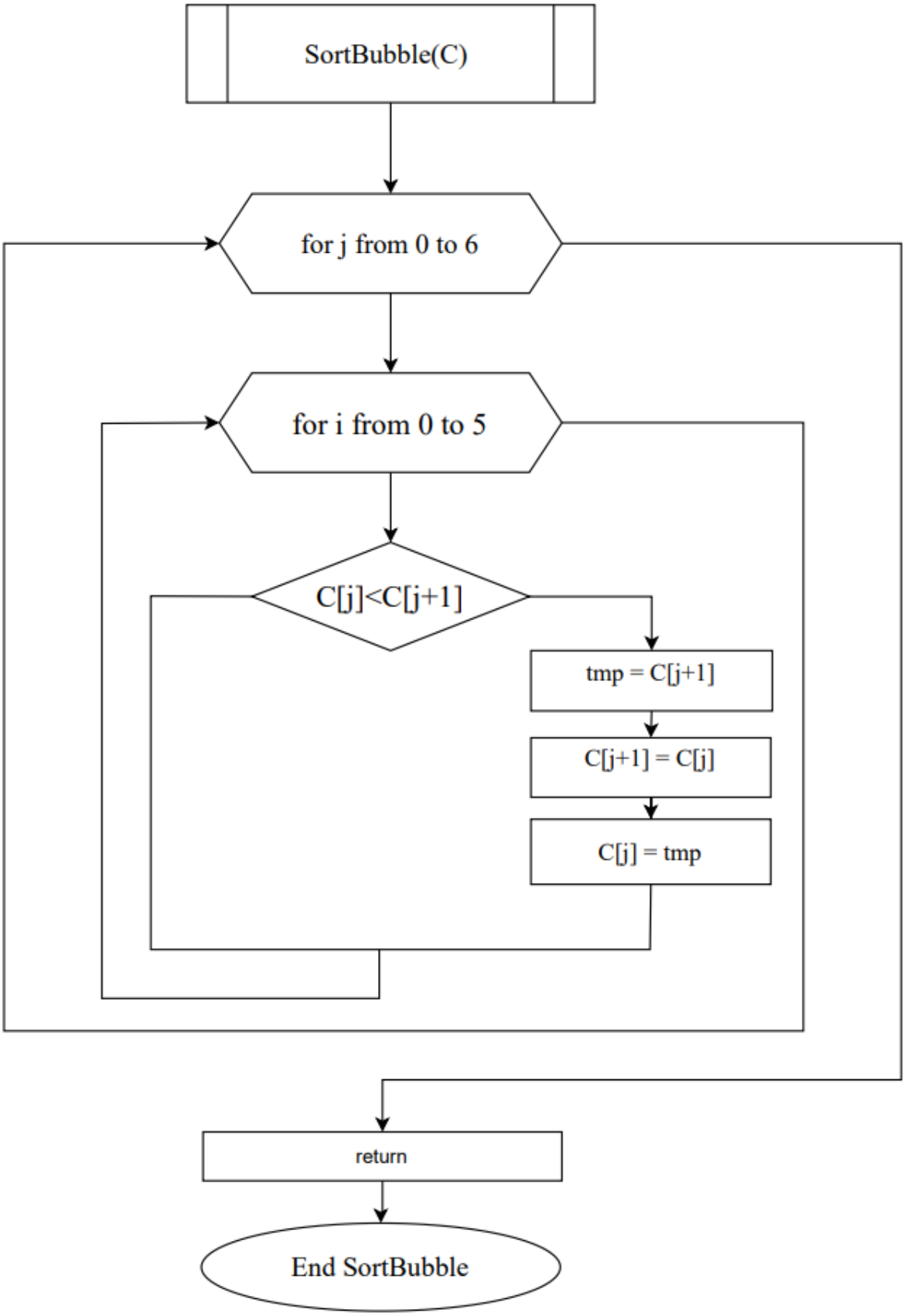

Start
A=CreateArrayTwoDimensional()
C=CreateArrayOneDimensional(A)
CoutArrayTwoDimensional(A)
CoutArrayOneDimensional(C)
C=SortBubble(C)
CoutArrayOneDimensional(C)
End

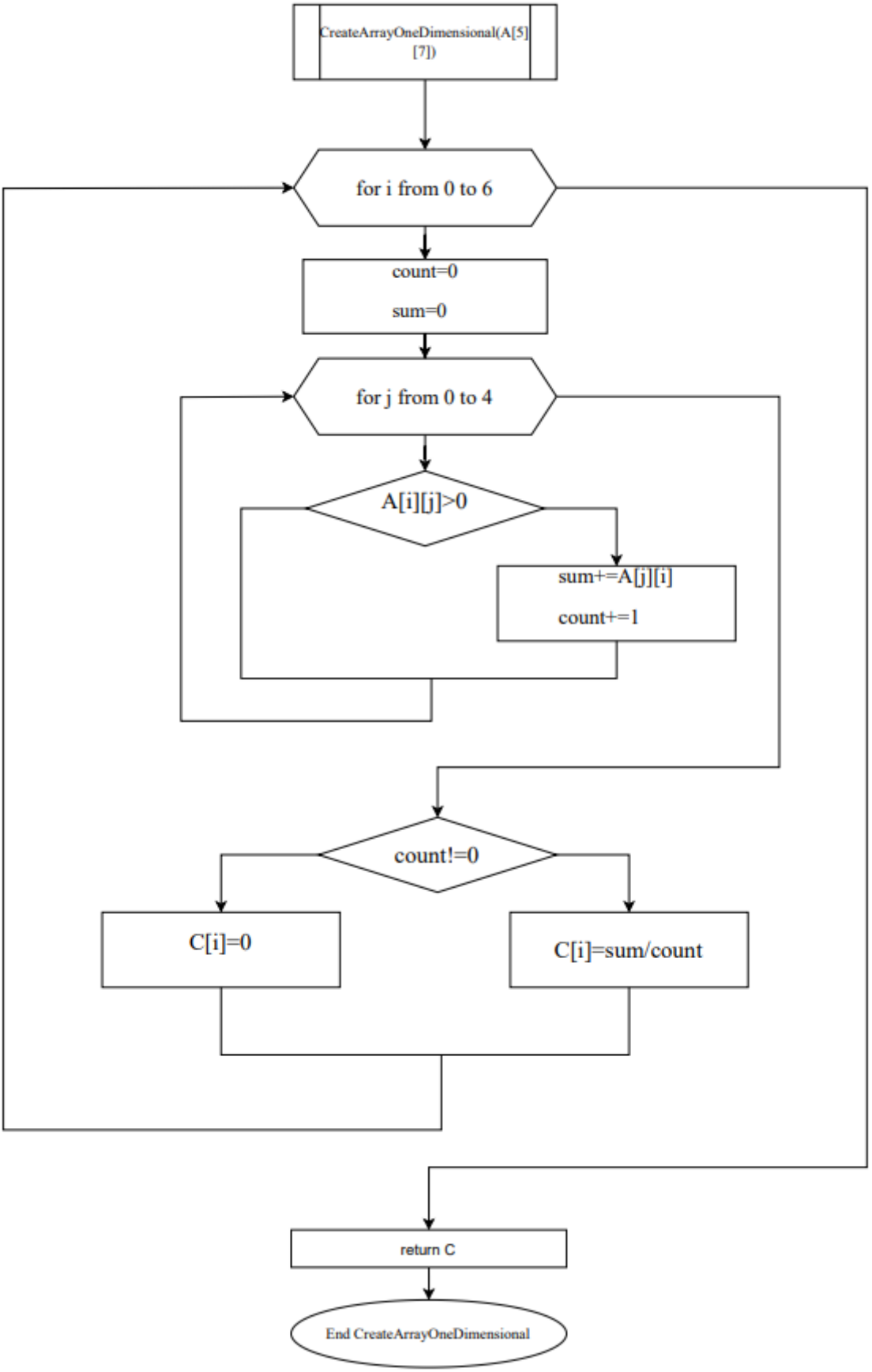
Підпрограми




CoutArrayOneDimensional(C)
for i from 0 to 6
output C[i]
return
End CoutArrayOneDimensional


CoutArrayTwoDimensional(A[5][7])
for j from 0 to 4
for i from 0 to 6
output A[j][i]
return
End CoutArrayTwoDimensional


SortBubble(C)
for j from 0 to 6
for i from 0 to 5
C[j]<C[j+1]
tmp = C[j+1]
C[j+1] = C[j]
C[j] = tmp
return
End SortBubble


CreateArrayOneDimensional(A[5][7])
for i from 0 to 6
count=0
sum=0
for j from 0 to 4
A[i][j]>0
sum+=A[j][i]
count+=1
count!=0
C[i]=0
C[i]=sum/count
return C
End CreateArrayOneDimensional

Код на C++:

```
#include <iostream>
#include <ctime>
#include <iomanip>
using namespace std;
void CreateArrayOneDimensional(double[5][7], double*);
void CreateArrayTwoDimensional(double[5][7]);
void CoutArrayTwoDimensional(double[5][7]);
void CoutArrayOneDimensional(double*);
void SortBubble(double *);

int main() {

    srand(time(NULL));
    double C[7];
    double A[5][7];
    CreateArrayTwoDimensional(A);
    CoutArrayTwoDimensional(A);

    CreateArrayOneDimensional(A, C);
    cout << "\tOne-Dimensional Array before sorting:\n\n ";
    CoutArrayOneDimensional(C);
    SortBubble(C);
    cout << "\tOne-Dimensional Array after sorting:\n\n ";
    CoutArrayOneDimensional(C);
```

```
    return 0;
}


void SortBubble(double* C) {
    double tmp;
    for (int i = 0; i < 7; i++) {
        for (int j = 0; j < 6; j++)
            if(C[j]<C[j+1]){
                tmp = C[j+1];
                C[j+1] = C[j];
                C[j] = tmp;
            }

    }
    return;
}

void CoutArrayOneDimensional(double* A) {
    cout << "[";
    for (int i = 0; i < 7; i++) {
        cout << setw(4) << A[i] << ((i == 6) ? "]\n\n" : "\t");
    }
    return;
}
```

```
void CreateArrayOneDimensional(double A[5][7], double* C) {
	double sum;
	int count;

	for (int i = 0; i < 7; i++) {
		count = 0;
		sum = 0;
		for (int j = 0; j < 5; j++)
		{

			if (A[j][i] > 0) {
				sum += A[j][i];
				count++;
			}
		}
		(count!=0) ? (C[i] = sum / float(count)) : (C[i]=0);

	}
	return;
}

void CoutArrayTwoDimensional(double A[5][7]) {
	cout << "\t\tTwo-dimensional Array 5x7:\n" << endl;

	for (int j = 0; j < 5; j++) {
		for (int i = 0; i < 7; i++) {
```

```
                cout << setw(4) << A[j][i] << ((i == 6) ? "\n\n" :
"\t");
            }
        }

        return;
}


void CreateArrayTwoDimensional(double A[5][7]) {

        for (int j = 0; j < 5; j++) {
            for (int i = 0; i < 7; i++) {
                A[j][i] = (double(rand() % 2000) - 1000) / 10.0;

            }
        }
        return;
}
```

Копії екранних форм:

```cpp
#include <iostream>
#include <ctime>
#include <iomanip>
using namespace std;
void CreateArrayOneDimensional(double[5][7], double*);
void CreateArrayTwoDimensional(double[5][7]);
void CoutArrayTwoDimensional(double[5][7]);
void CoutArrayOneDimensional(double*);
void SortBubble(double *);

int main() {

    srand(time(NULL));
    double C[7];
    double A[5][7];
    CreateArrayTwoDimensional(A);
    CoutArrayTwoDimensional(A);

    CreateArrayOneDimensional(A, C);
    cout << "\tOne-Dimensional Array before sorting:\n\n ";
    CoutArrayOneDimensional(C);
    SortBubble(C);
    cout << "\tOne-Dimensional Array after sorting:\n\n ";
    CoutArrayOneDimensional(C);

    return 0;
}


void SortBubble(double* C) {
    double tmp;
    for (int i = 0; i < 7; i++) {
        for (int j = 0; j < 6; j++)
            if(C[j]<C[j+1]){
                tmp = C[j+1];
                C[j+1] = C[j];
                C[j] = tmp;
            }

    }
    return;
}

```

```
44  void CoutArrayOneDimensional(double* A) {
45      cout << "[";
46      for (int i = 0; i < 7; i++) {
47          cout << setw(4) << A[i] << ((i == 6) ? "]\n\n" : "\t");
48      }
49      return;
50  }
51
52  void CreateArrayOneDimensional(double A[5][7], double* C) {
53      double sum;
54      int count;
55
56      for (int i = 0; i < 7; i++) {
57          count = 0;
58          sum = 0;
59          for (int j = 0; j < 5; j++)
60          {
61
62              if (A[j][i] > 0) {
63                  sum += A[j][i];
64                  count++;
65              }
66          }
67          (count!=0) ? (C[i] = sum / float(count)) : (C[i]=0);
68
69      }
70      return;
71  }
72
73  void CoutArrayTwoDimensional(double A[5][7]) {
74      cout << "\t\tTwo-dimensional Array 5x7:\n" << endl;
75
76      for (int j = 0; j < 5; j++) {
77          for (int i = 0; i < 7; i++) {
78
79              cout << setw(4) << A[j][i] << ((i == 6) ? "\n\n" : "\t");
80          }
81      }
82
83      return;
84  }
85
```

```
86
87  void CreateArrayTwoDimensional(double A[5][7]) {
88
89      for (int j = 0; j < 5; j++) {
90          for (int i = 0; i < 7; i++) {
91              A[j][i] = (double(rand() % 2000) - 1000) / 10.0;
92
93          }
94      }
95      return;
96  }
```

```
                Two-dimensional Array 5x7:
14.1     7.8    -12.7   -22.4   24.9    82.9    93.3
-34.8   -59.3   -53.3   51.3    30.7     -8     -88.1
-96.2   -9.9      62     4.4      31     -92     -79.8
-42.9   38.7    -89.3   22.9    -19.8   -47.2   -99.8
-49.3   -88.4   -86.4    8.9    37.1    86.6    69.1
        One-Dimensional Array before sorting:
 [14.1  23.25     62    21.875  30.925  84.75   81.2]
        One-Dimensional Array after sorting:
 [84.75 81.2      62    30.925  23.25   21.875  14.1]
```

```
                Two-dimensional Array 5x7:
20.3     -31    -38.7   22.7    -38.2   -9.9    53.4
 -43     -55.3   95.7   -29.7   -85.6   57.5    -18.1
70.5     -35.5   64.2    44.3    32.5   -70.3     72
79.7     75.7   -68.8   -45.8   -68.9   40.7    -87.2
76.5     -78.3   61.8   -29.7    91.7    0.9    -88.5
        One-Dimensional Array before sorting:
 [61.75 75.7    73.9    33.5    62.1    33.0333 62.7]
        One-Dimensional Array after sorting:
 [75.7  73.9    62.7    62.1    61.75   33.5    33.0333]
```

Випробування алгоритму

Проведемо випробування на прикладі другого результату

Блок	Дія
	Початок
1	Перевірка утворення одновимірного масиву: C[0]=(20.3+70.5+79.7+76.5)/4=61.75 C[1]=(75.7)/1=75.7 C[2]=(95.7+64.2+61.8)/3=73.9 C[3]=(22.7+44.3)/2=33.5 C[4]=(32.5+91.7)/2=62.1 C[5]=(57.5+40.7+0.9)/3=33.0(3) C[6]=(53.4+72)/2=62.7
2	Сортування(перевіряємо з результатом) 75.7 73.9 62.7 62.1 61.75 33.5 33.0(3)
	Кінець

Висновок

Отже, виконавши цю лабораторну роботу, ми навчилися використовувати сортування в масивах(на прикладі сортування обміном). Ідея цього сортування полягає в тому, щоб поступово порівнювати значення двох сусідніх елементів і в тому разі, якщо поточний, наприклад, більше за наступний, то ми його рухаємо вперед. Такий алгоритм можна реалізувати через два арифметичних цикли, бо нам потрібно повністю пробігти по масиву n разів(n – довжина масиву). Після кожного повного пробігу найбільший елемент встає на своє місце. У ході виконання роботи створюємо двовимірний масив 5x7, який заповнюємо дійсними числами. Потім утворюємо одновимірний масив з середнього арифметичного значення додатних елементів кожного стовпчика. Якщо ж усі елементи від’ємні, то записуємо нуль, інакше мали б ситуацію 0/0(sum/count). Після цього сортуємо цей одновимірний масив та виводимо його. У процесі виконання ми сформулювали задачу, побудували математичну модель та псевдокод алгоритму, що допомогло нам краще її зрозуміти