Агреговані типи даних. Статичні масиви

Лектор:
к.т.н., доцент кафедри прикладної математики
Рижа Ірина Андріївна

Про що ця лекція???

- Наведемо особливості оголошення та ініціалізації одновимірного масиву.
- Опишемо схему розташування масивів у пам'яті.
- Розглянемо способи заповнення елементів масиву.

Оголошення статичного масиву

Масив

— це впорядкована сукупність однотипних даних.

- Кожен елемент масиву має свій порядковий номер у цій сукупності.

Синтаксис оголошення статичного масиву

```
тип_елемента_масиву ідентифікатор_масиву[розмірність];
```

- `тип_елемента_масиву` – будь-який зі стандартних типів мови C/C++ (крім void), чи тип користувача, в тому числі масив.
- `ідентифікатор_масиву` (ім'я масиву) – унікальна для даного блоку послідовність із латинських літер, цифр чи символу підкреслення, що не починається цифрою.
- `розмірність` повинна мати константний характер, тобто повинна задаватися **цілочисельною додатною константою** (абсолютно чи поіменованою).

Оголошення статичного масиву

Приклади оголошення статичного масиву

```
int masuv_int[10];
double masuv_real[20];
char ar_ch[15];.
```

- Якщо масив оголошено як **глобальний**, тобто поза межами будь-якого блоку (зокрема функції `main()`), то початкове значення його елементів є визначеним і рівним нулю відповідного типу.
- Якщо ж масив є **локальним**, тобто оголошений всередині певного блоку, то він вважається неініціалізованим, і початкове значення його елементів є невизначеним, тобто «сміттєвим».

Оголошення статичного масиву

При оголошенні масиву НЕ виконується неявне приведення типу.

- Розмірність повинна бути ненульової та цілочисельною.
- Задання розмірності константою невідповідного типу чи значення призводить до синтаксичної помилки.

Наприклад,

```
int a[12.5]; //недопустимо
int b['A'];  //допустимо
```

Задання поіменованих констант

Способи задання поіменованих констант:

1. Оголошення цілочисельної константи з використанням модифікатора `const`:

```
const int size1 = 10;
int masuv[size1];
```

2. Використання директиви препроцесора `#define`, за допомогою якої означується константне поіменоване значення.

```
...
#define size1 10
int masuv [size1];
int main()
{...
```

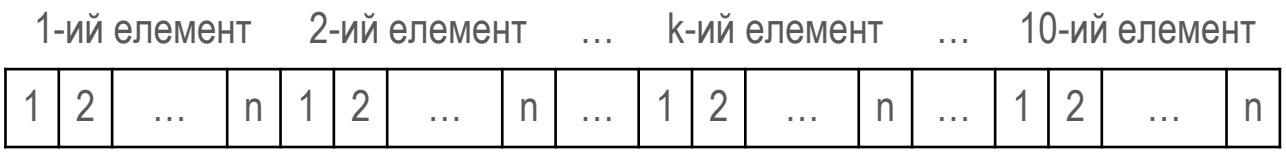
3. Використання використано перелічуваного типу `enum`.

```
enum { size1 = 10, size2 = 20 };
int masuv_int[size1];
```

Розташування елементів масиву у пам'яті

- Елементи масиву займають **суцільновиділену** (неперервну) область оперативної пам’яті.
- Кожен елемент займає стільки байт, скільки визначено програмним середовищем для зберігання відповідного типу.
- Розмір області оперативної пам’яті дорівнює добутку кількості елементів на обсяг пам’яті, що його займає одне дане відповідного типу.

```
int b[10];    //10x4 байти
double c[10]; //10x8 байтів
```



Ініціалізація елементів масиву

Для того, щоби ініціалізувати елементи масиву, після його оголошення ставимо знак дорівнює і тоді у фігурних дужках через кому перераховуємо список відповідних значень:

```
тип_елемента_масиву ідентифікатор[розмірність] = {список значень};
```

- Дані зі списку повинні бути відповідного типу.
- Кількість перерахованих значень НЕ повинна перевищувати задану розмірність масиву.
- Якщо кількість значень у списку є меншою, ніж вказана розмірність масиву, то елементи, яким не вистачило значення стають рівними нулеві того типу, якого є елементи масиву.

```
...
const int size1 = 10;
int main()
{
     int b[size1] = {0,-12,4,1,7,4,6,3,5,2};
     float c[size1] = {0}; //занулюються всі елементи масиву
```

Ініціалізація елементів масиву

При оголошенні масиву з одночасною ініціалізацією його елементів дозволяється не вказувати розмірність оголошуваного масиву.

- Комп'ютер виділить такий розмір пам'яті, який є необхідним для зберігання заданого списку значень.

```
int a[] = { 1,-1,2,-2,3,4,5,6,7,8,9 };
```

Операції над масивами

Операція `sizeof`

встановлює розмір в байтах оперативної пам'яті, яку займає масив.

```
sizeof (ідентифікатор);
```

або

```
sizeof ідентифікатор;
```

Наприклад,

```
int a[] = { 1,-1,2,-2,3,4,5,6,7,8,9 };
int k = sizeof a; // k=44
```

Операції над масивами

Операція взяття елементу масиву

Здійснює доступ до елемента масиву за його порядковим номером (індексом).

```
ім’я_масиву[індекс_елемента_масиву];
```

- Індекс першого елемента дорівнює нулю, а індекс останнього на одиницю менший, ніж розмірність масиву.

Компілятор НЕ відслідковує правильності використання індексації в масивах.

Заповнення масиву

Способи заповнення масиву:

1. при оголошенні з одночасною ініціалізацією;
2. введення даних з клавіатури;
3. за певним законом з використанням математичних функцій;
4. генерування псевдовипадкових чисел.

Заповнення масиву

Введення даних з клавіатури

Наприклад,

```
...
const int size1 = 10;
double a[size1] = { 0.0 };
...
for (int i = 0; i < size1; i++)
{
     cout << "a[" << i << "] = ";
     cin >> a[i];
}
...
```

Заповнення масиву

За певним законом з використанням математичних функцій

Наприклад,

```
#include<cmath>
...
const int size1 = 10;
double a[size1] = { 0.0 };
...
for (int i = 0; i < size1; i++)
{
     a[i] = 10.0*sin(double(i));
}
...
```

Заповнення масиву

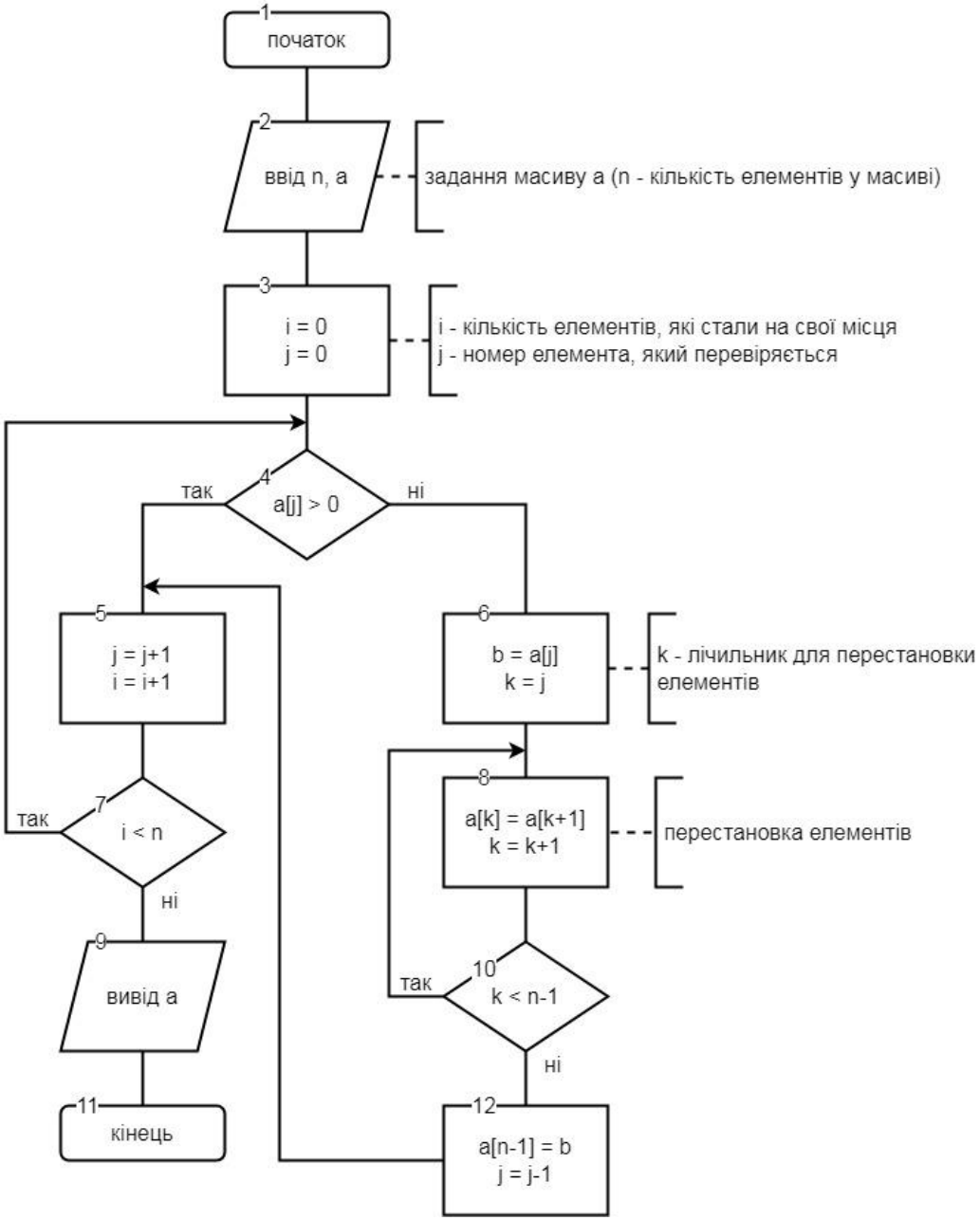
Генерування псевдовипадкових чисел

Наприклад,

```
#include<stdlib.h>
#include<time.h>
...
const int size1 = 10;
double a[size1] = { 0.0 };
...
srand(time(0));
for (int i = 0; i < size1; i++)
{
     a[i] = rand();
}
...
```

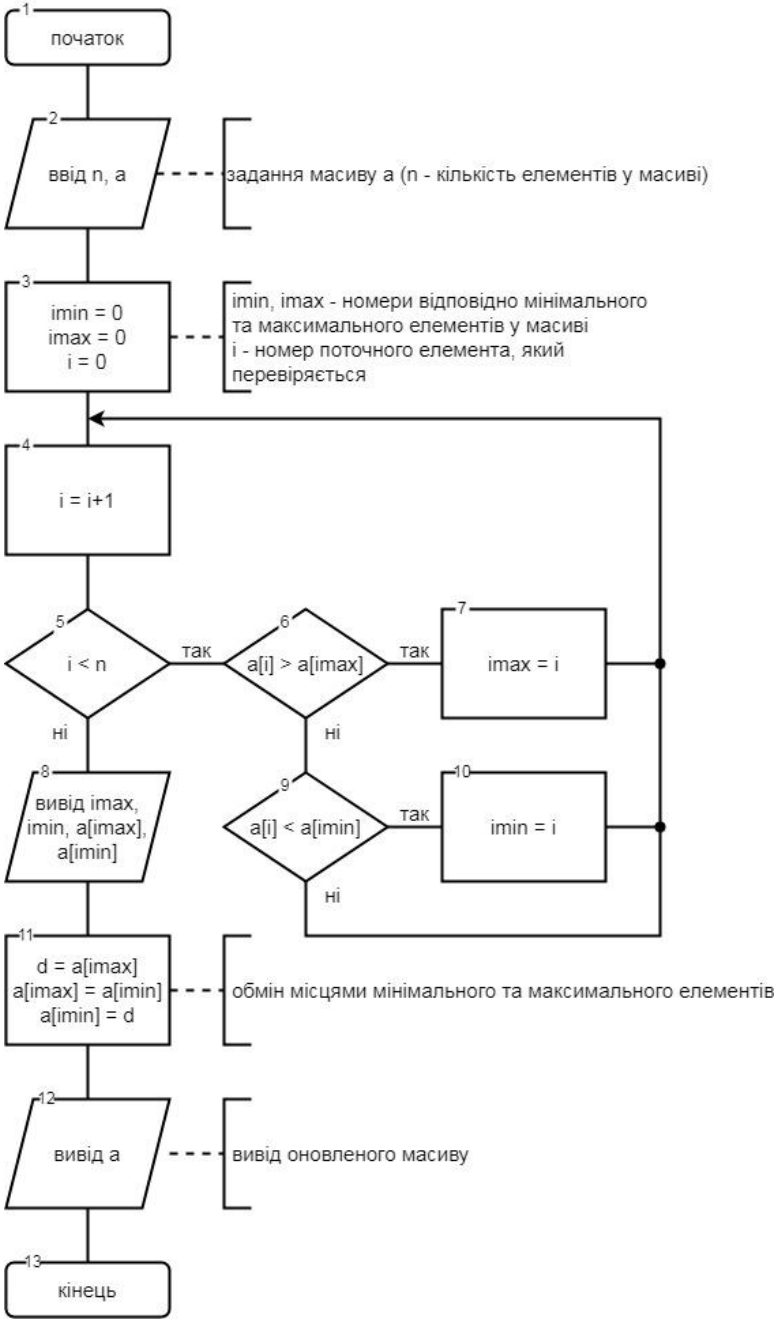
Приклад 1

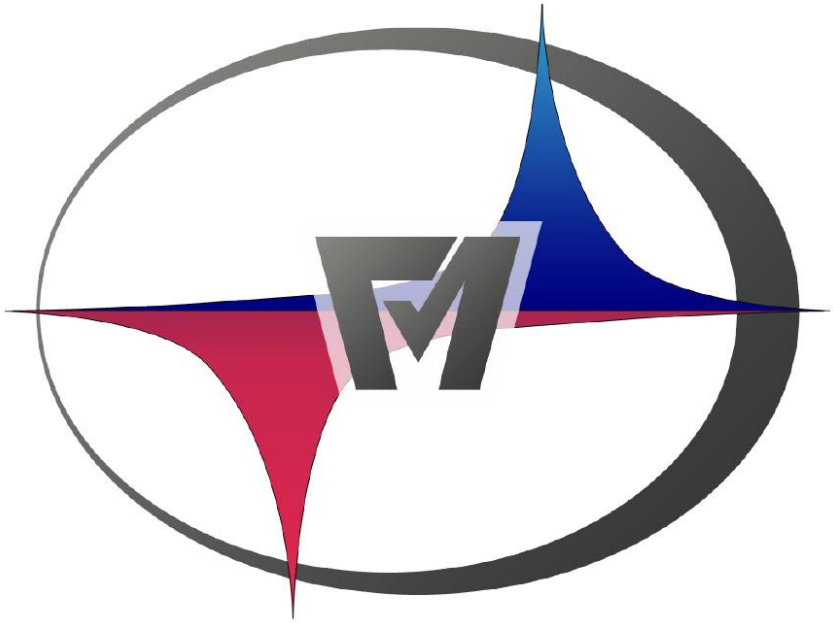
Впорядкувати заданий масив цілих чисел наступним чином: спочатку у масиві повинні бути всі додатні елементи, потім недодатні. Порядок слідування у результуючому масиві повинен бути збережений. Додаткового масиву не використовувати.


1
початок
2
ввід n, a
задання масиву a (n - кількість елементів у масиві)
3
i = 0
j = 0
i - кількість елементів, які стали на свої місця
j - номер елемента, який перевіряється
4
a[j] > 0
так
ні
5
j = j+1
i = i+1
6
b = a[j]
k = j
k - лічильник для перестановки елементів
7
i < n
так
ні
8
a[k] = a[k+1]
k = k+1
перестановка елементів
9
вивід a
10
k < n-1
так
ні
11
кінець
12
a[n-1] = b
j = j-1

Приклад 2

Знайти координати найбільшого та найменшого елементів масиву дійсних чисел і поміняти їх місцями.


1
початок
2
ввід n, a
задання масиву a (n - кількість елементів у масиві)
3
imin = 0
imax = 0
i = 0
imin, imax - номери відповідно мінімального
та максимального елементів у масиві
i - номер поточного елемента, який
перевіряється
4
i = i+1
5
i < n
так
6
a[i] > a[imax]
так
7
imax = i
ні
ні
8
вивід imax,
imin, a[imax],
a[imin]
9
a[i] < a[imin]
так
10
imin = i
ні
11
d = a[imax]
a[imax] = a[imin]
a[imin] = d
обмін місцями мінімального та максимального елементів
12
вивід a
вивід оновленого масиву
13
кінець



Кафедра прикладної математики

http://amath.lp.edu.ua