ЛЕКЦІЯ 3

1. БАЗОВІ ТИПИ ДАНИХ. КОНСТАНТИ
2. ПРАВИЛА ВИДИМОСТІ
3. БЛОК-СХЕМИ. ГРАФІЧНИЙ СПОСІБ ОПИСУ АЛГОРИТМІВ
4. АЛГОРИТМ ВИЗНАЧЕННЯ МІНІМАЛЬНОГО ПОЗИТИВНОГО ЕЛЕМЕНТУ МАСИВУ

БАЗОВІ ТИПИ ДАНИХ

У мові С усі змінні мають бути описані перед їхнім використанням. У неї визначено 5 типів даних, які можливо назвати базовыми:

char -символьний,
int - цілий,
float - дійсний,
double - дійсний подвійної довжини,
void - пустий, той, що не має значення.

Типи char та int є цілочисельними, вони призначені для зберігання цілих чисел.

char (character) – символьна змінна. Будь-який символ пов'язаний з цілим числом – кодом цього символу, наприклад, в таблиці ASCII. Символ відтворюється на екрані за його кодом, при введенні з клавіатури символ перетворюється у відповідне значення. Такі перетворення відбуваються автоматично. Тип char за замовчуванням є знаковим типом, однак налаштуванням опцій інтегрованого середовища можливо це змінити на беззнаковий тип char. Фрагмент таблиці з кодами ASCII

48	**060**	**0x30**	**00110000**	**0**			**Zero**
49	061	0x31	00110001	1			One
50	062	0x32	00110010	2			Two
51	063	0x33	00110011	3			Three

Тип int завжди знаковий, так само, як float и double.

Тип void додано до мови для реалізації додаткових можливостей.

На основі 5 базових типів будуються похідні.

Такі типи будуються за допомогою модификаторів (modifiers) типу, які додаються перед відповідним типом.
У стандарті ANSI мови С такими модифікаторами є наступні зарезервовані слова:

signed -знаковий,
unsigned - беззнаковий,
long - довгий,
short -короткий.

Модифікаторы signed та unsigned можуть застосовуватись до типів char та int.
Модифікатори short та long можуть застосовуватись до типу int.
Модифікатор long може застосовуватись також до типу double.
Модифікатори signed та unsigned можуть комбінуватись з модифікаторами short та long стосовно типу int.
Всі типи мови представлено в таблиці.

Тип	Розмір у байтах (бітах)	Інтервал змінювання	
char	1 (8)	від -128	до 127
unsigned char signed char int	1 (8) 1 (8) 2 (16)	від 0 від-128 від-32768	до 255 до 127 до 32767
unsigned int signed int short int	2 (16) 2 (16) 2 (16)	від 0 від-32768 від-32768	до 65535 до 32767 до 32767

unsigned short int	2 (16)	від 0	до 65535
signed short int	2 (16)	від -2147483648	до 32767
long int	4 (32)	від -2147483648	до 2147483647
signed long int	4 (32)	від -2147483648	до 2147483647

unsigned long int	4 (32)	від 0	до 4294967295
float	4 (32)	від 3.4E-38	до 3.4E+38
double	8 (64)	від 1.7E-308	до 1.7E+308
long double	10 (80)	від 3.4E-4932	до 3.4E+4932

P.S.В останніх реалізаціях мови можуть бути зміни, наприклад цілочисельний тип може займати 4 байти

Константи в мові С

- фіксовані величини, що не можуть бути змінені у програмі. Константи можуть бути будь-якого базового типу даних. Приклади констант:

Тип даних	*Константа*
char	'a', '\n', '9'
int	1, 123, -346
unsigned int	60000
long int	75000, -275
short int	12, -128
float	123.23, 4.34E-3, 4E+5
double	123.23, 1.2312311, -0.987

До якого типу відноситься константа 13 - типу char, int, unsigned або іншому?
Правила визначення типу констант наступні :
Ціла константа (тобто така, що не має десяткової точки або порядку) відноситься до int, якщо вона входить до інтервалу значень типу int. Якщо не входить в интервал значень типу int, наприклад 37000, то вона вважається константой типу unsigned. Якщо ж константа не входить в інтервал змінювання unsigned, вона вважається константой типу long. Також в різних версіях можуть використовуватись індекси, що безпосередньо вказують на тип констнти u (unsigned), l(long), h (short), f (float).
Шістнадцяткові константи мають префікс 0x, восьмірічні 0.

Константа з десятковою точкою вважається константою типу double, якщо вона відповідає відповідному інтервалу змінювання.

ОГОЛОШЕННЯ (ОПИС) ЗМІННИХ

тип < список змінних через кому>;

Наприклад int x; double y; long double integral; …

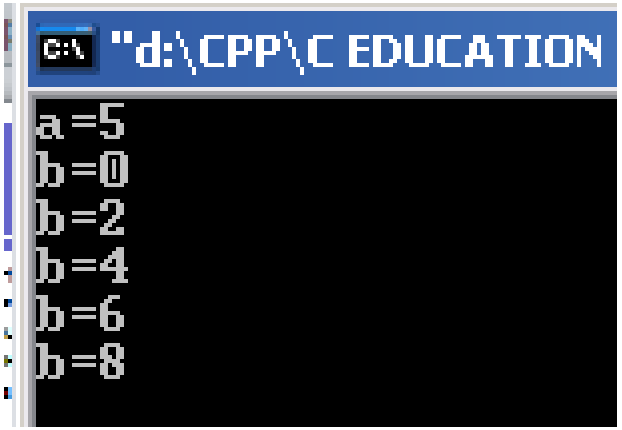
При описі змінної велике значення має місце, де її оголошено.
Правило, яке визначає, де змінні можуть бути описані, називають правилом видимості (scope rule).
Змінні можуть бути описані:

- поза будь-якою функцією, в тому числі й main() - глобально;
- всередині блока (функції) – локально;
- як параметр функції – локально.

Локально описані змінні можуть використовуватись тільки в тому блоці, де їх описано!

main.cpp

```
1   #include <iostream>
2   using namespace std;
3   int main()
4   {int i, a; a=5;
5   cout<<"a="<<a<<"\n";
6
7      for(i=0;i<5;i++)
8      {
9           int b;
10          b=2*i;
11          cout<<"b="<<b<<"\n";
12     }
13
14        return 0;
15  }
```



main.cpp

```
1   #include <iostream>
2   using namespace std;
3   int main()
4   {int i, a; a=5;
5   cout<<"a="<<a<<"\n";
6
7      for(i=0;i<5;i++)
8      {
9           int b;
10          b=2*i;
11          cout<<"b="<<b<<"\n";
12     }
13   cout<<"b="<<b<<"\n";
14       return 0;
15  }
```

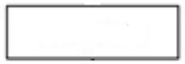
File	Line	Message
d:\CPP\C EDUCA...		In function 'int main()':
d:\CPP\C EDUCA...	13	error: 'b' was not declared in this scope
		=== Build failed: 1 error(s), 0 warning(s) (

БЛОК-СХЕМИ. ГРАФІЧНИЙ СПОСІБ ОПИСУ АЛГОРИТМІВ

На практиці частіше зустрічається інший спосіб опису алгоритмів – графічний.
Кожен елемент алгоритму представляється певним графічним примитивом. Загальний хід програми з початку до кінця відображається лініями зі стрілками. Для відораження основних елементів алгоритмів використовується наступні графічні елементи:

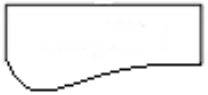
Початок алгоритму

Оператор



Умовний оператор

Введення даних 

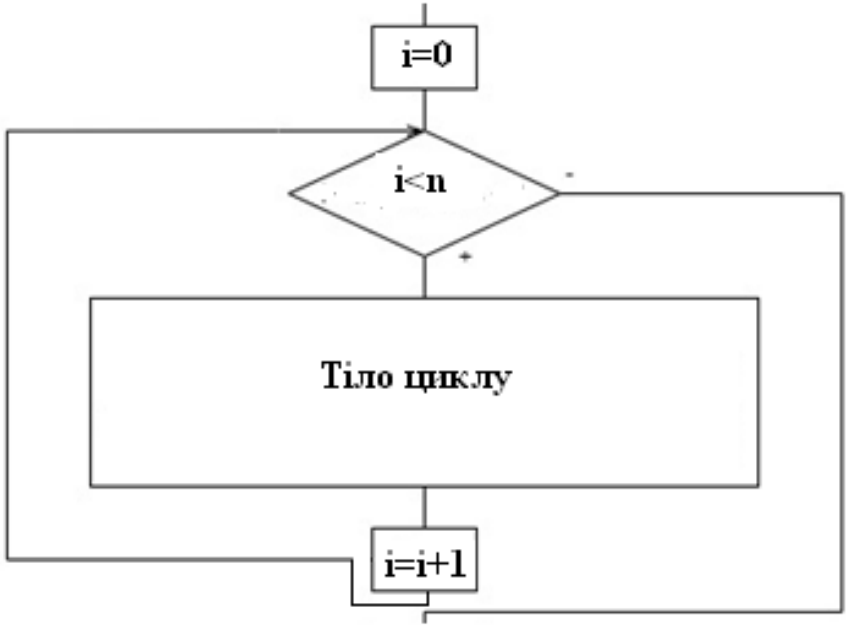
Виведення результатів 

Підпрограма 

Кінець програми/ алгоритму 

Якщо алгоритм займає більше однієї сторінки, для переходу на іншу використовуються символи (літери, літери з цифрами), розміщені у колі на обох сторінках.

Приклад. Оператор циклу for (i=0;i<n;i=i+1)

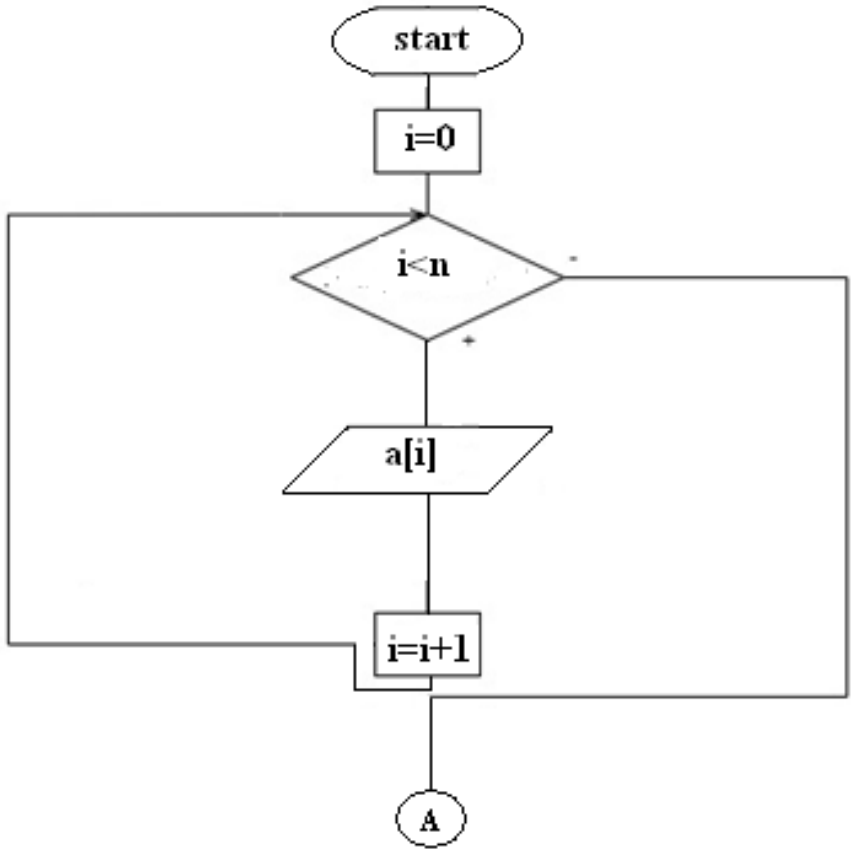


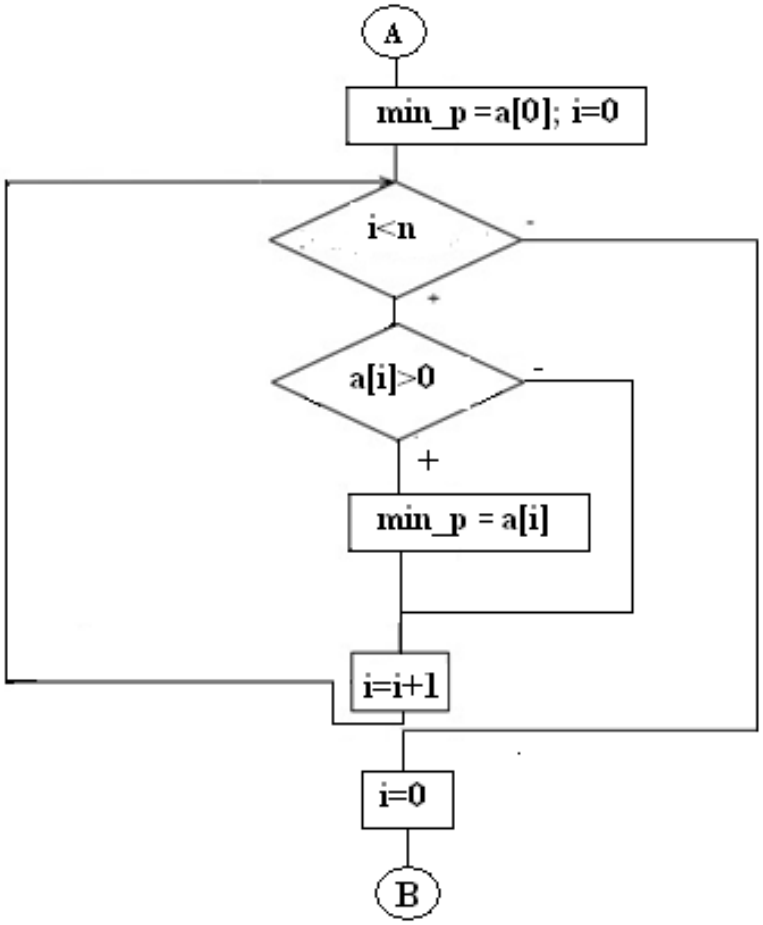
Приклад.

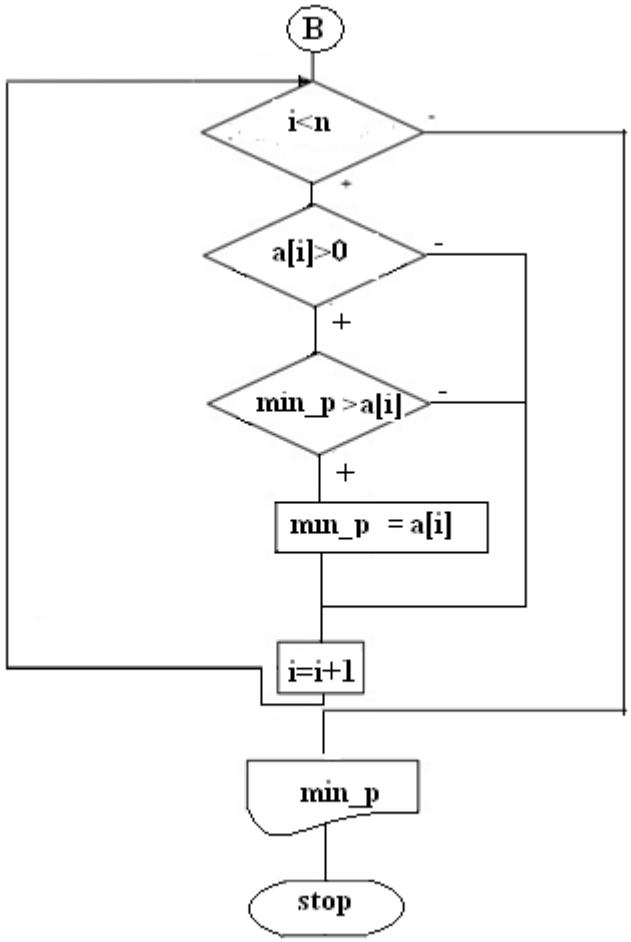
Написати алгоритм та програму, в якій для заданого користувачем одновимірного масиву визначається мінімальний позитивний його елемент. Вважається (для початку), що хоча б один позитивний елемент в масиві є.

Увага! Важлива частина алгоритму – пошук початкового значення для мінімального позитивного елементу. Воно має бути вибране з позитивних елементів масиву!

(n=4)




A
min_p =a[0]; i=0
i<n
-
+
a[i]>0
-
+
min_p = a[i]
i=i+1
i=0
B


B
i<n
a[i]>0
min_p >a[i]
min_p = a[i]
i=i+1
min_p
stop
-
+
-
+

*main.cpp

```
1   #include <iostream>
2   using namespace std;
3   int main()
4   {int i, a[4], min_p;
5   /** Пошук мінімального позитивного елементу масиву a*/
6
7      for(i=0;i<4;i++)
8      cin>>a[i];
9      min_p=a[0];
10     for(i=0;i<4;i++)
11     if (a[i]>0) min_p=a[i];
12
13      for(i=0;i<4;i++)
14      if (a[i]>0)
15          if (min_p>a[i]) min_p=a[i];
16
17
18     cout<<"Minimum positive value of array elements="<<min_p;
19      return 0;
20  }
```

```
"d:\CPP\C EDUCATION 1 sem 1 year\c3\bin\Debug\p2019 1.exe"
20
-3
1
8
Minimum positive value of array elements=1
Process returned 0 (0x0)   execution time : 12.828 s
Press any key to continue.
```

ДЯКУЮ ЗА УВАГУ!