

# Реалізація алгоритмів з розгалуженням

## Перетворення у рядок (String)

Рядкове перетворення відбувається, коли вимагається подання чого-небудь у вигляді рядка (наприклад, коли відбувається виведення даних за допомогою **alert**: **alert(25)** => **alert('25')**).

Способи перетворення	Приклад
<code>String(val);</code>	<code>let s1 = String(27) //s1 = '27'</code>
<code>"...рядок..." + значення</code>	<code>let s2 = 'Age = ' + 27 //s2 = 'Age = 27'</code>
З використанням методів: <code>toFixed</code> <code>toPrecision</code> <code>toString</code> <code>toLocaleString</code> . . . . .	<code>2.34.toFixed(1); // '2.3'</code> <code>245.4589.toPrecision(2) // '2.5e+2'</code> <code>245.4589.toString() // '245.4589'</code> <code>245.4589.toLocaleString('uk-UA') // '245,459'</code>

## Перетворення у число (*Number*)

Чисельне перетворення відбувається в математичних функціях і виразах, а також при нестрогому порівнянні даних різних типів.

Для перетворення до числа в явному вигляді можна:

- викликати `Number(val)` ;
- покласти перед виразом оператор `+` ;
- `parseInt(рядок_з_цілим_числом)` ;
- `parseFloat(рядок_з_дійсним_числом)` ;

Значення	Перетвориться в ...
undefined	NaN
null	0
true	1
false	0
Рядок	Пробіли по краях обрізаються. Далі один з випадків:  1) якщо залишається порожній рядок, то 0 ;  2) з непорожнього рядка зчитується число;  3) якщо у рядку не число, то результат NaN .

## Логічний тип

Логічний тип даних `Boolean` має всього два значення:

- `true` («істина», «вірно» або ж «так»);
- `false` («неправда», «невірно» або ж «ні»).

### *Перетворення до логічного типу даних (`Boolean`)*

Перетворення до логічного типу (до `true/false`) відбувається у логічному контексті (там де необхідно перевірити виконання умови), такому як `if(obj)` , `while(obj)` і при застосуванні логічних операторів.

Значення	Перетвориться в ...
<code>undefined, null</code>	<code>false</code>
Числа	Всі <code>true</code> , крім <code>0</code> , <code>NaN</code> - <code>false</code> .
Рядки	Всі <code>true</code> , крім порожнього рядка <code>" "</code> - <code>false</code>
Об'єкти	Завжди <code>true</code>

## Логічні вирази

*Логічний вираз* -- це вираз в результаті одержуємо одне із логічних значень (`true/false`).

Логічні вирази можуть містити:

- логічні константи (`true/false`);
- змінні логічного типу;
- оператори порівняння;

У JavaScript використовуються такі оператори порівняння

Більше	<code>a &gt; b</code>
Менше	<code>a &lt; b</code> .
Більше або дорівнює	<code>a &gt;= b</code>
Менше або дорівнює	<code>a &lt;= b</code>
Рівне ( <code>a=b</code> це присвоювання!)	<code>a == b</code>
Не рівно, $\neq$	<code>!=</code>
Тотожно рівно (співпадає тип і значення)	<code>a === b</code>
Не тотожно рівно (не співпадає тип, або не співпадають значення)	<code>A !== b</code>

Операції з логічними виразами

x	y	x&&y (логічне «і»)	x    y (логічне «або»)	!x (заперечення)
false	false	false	false	true
false	true	false	true	true
true	false	false	true	false
true	true	true	true	false

# Умовний оператор

Іноді, залежно від певних умов, потрібно виконати різні дії. Для цього використовується умовний оператор. Він може бути в повній та скороченій формах

Якщо (виконується умова)

То (виконуємо оператор\_1)

Інакше (виконуємо оператор\_2)

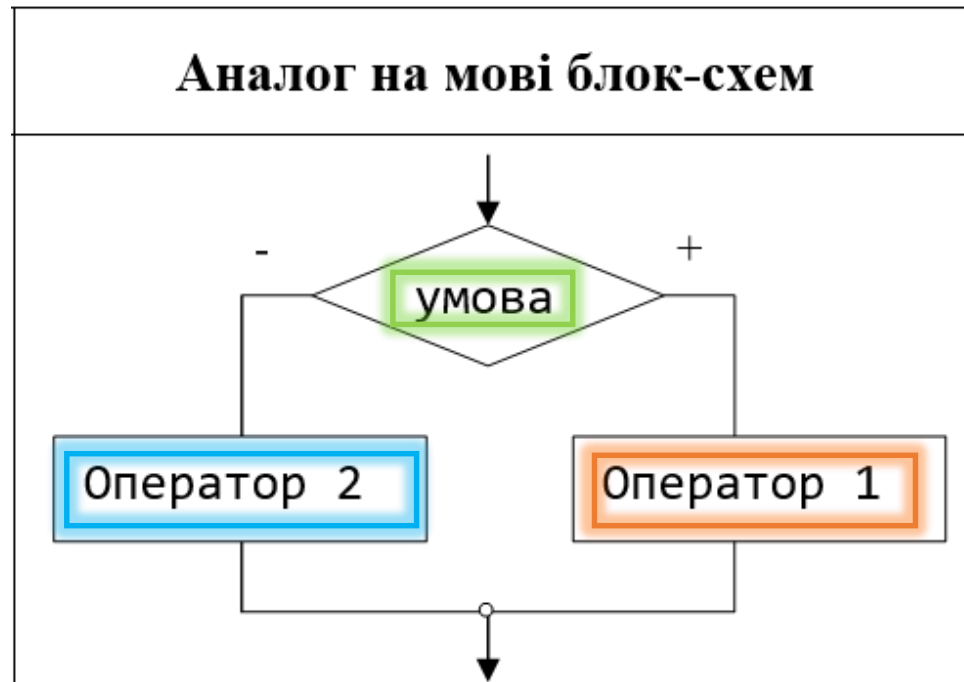
# Умовний оператор

Іноді, залежно від певних умов, потрібно виконати різні дії. Для цього використовується умовний оператор. Він може бути в повній та скороченій формах

Якщо (виконується умова)

То (виконуємо оператор\_1)

Інакше (виконуємо оператор\_2)





# Умовний оператор

Іноді, залежно від певних умов, потрібно виконати різні дії. Для цього використовується умовний оператор. Він може бути в повній та скороченій формах

Якщо (виконується умова)

То (виконуємо оператор\_1)

Інакше (виконуємо оператор\_2)

Програмна структура	Аналог на мові блок-схем
<pre>if (&lt;умова&gt;)     &lt;оператор1&gt; else     &lt;оператор2&gt;</pre>	<pre>graph TD; Entry(( )) --&gt; Decision{умова}; Decision -- "-" --&gt; Op2[Оператор 2]; Decision -- "+" --&gt; Op1[Оператор 1]; Op2 --&gt; Exit(( )); Op1 --&gt; Exit; Exit --&gt; Exit2(( ))</pre>

# Умовний оператор

Іноді, залежно від певних умов, потрібно виконати різні дії. Для цього використовується умовний оператор. Він може бути в повній та скороченій формах

Якщо (виконується умова)

То (виконуємо оператор\_1)

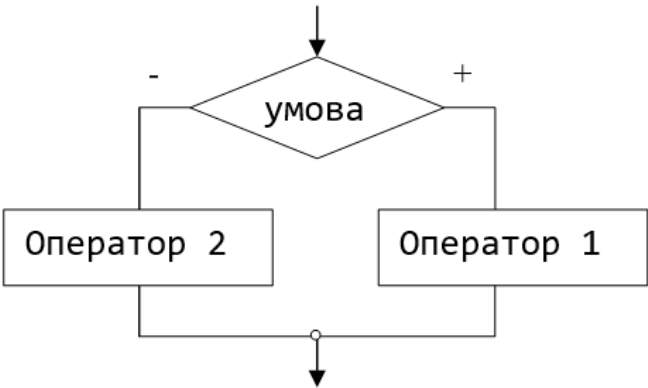
Інакше (виконуємо оператор\_2)

Програмна структура	Аналог на мові блок-схем	Приклад
<pre>if (&lt;умова&gt;)     &lt;оператор1&gt; else     &lt;оператор2&gt;</pre>	<pre>graph TD; Entry(( )) --&gt; Decision{умова}; Decision -- "-" --&gt; Op2[Оператор 2]; Decision -- "+" --&gt; Op1[Оператор 1]; Op2 --&gt; Exit(( )); Op1 --&gt; Exit; Exit --&gt; Exit2(( ))</pre>	<pre>if(x&gt;y)     max=x else     max=y</pre>

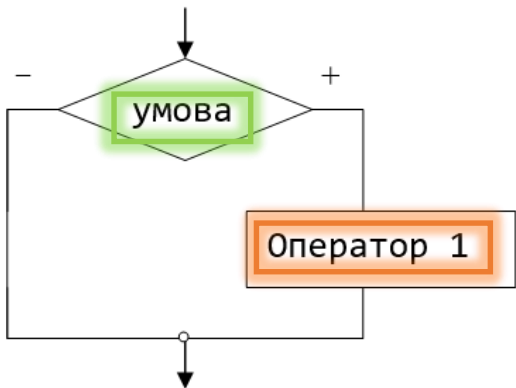
# Умовний оператор

Іноді, залежно від певних умов, потрібно виконати різні дії. Для цього використовується умовний оператор. Він може бути в повній та скороченій формах

## Повна форма

Програмна структура	Аналог на мові блок-схем	Приклад
<pre>if (&lt;умова&gt;)     &lt;оператор1&gt; else     &lt;оператор2&gt;</pre>		<pre>if(x&gt;y)     max=x else     max=y</pre>

## Скорочена форма

Програмна структура	Аналог на мові блок-схем	Приклад
<pre>if (&lt;умова&gt;)     &lt;оператор1&gt;</pre>		<pre>if(x!=0)     z=1/x</pre>

Приклад. На сайті банку задають питання стосовно віку користувача. Якщо вік  $<18$ , то повідомити про заборону доступу, інакше – привітати на сайті

Приклад. На сайті банку задають питання стосовно віку користувача. Якщо вік <18, то повідомити про заборону доступу, інакше – привітати на сайті

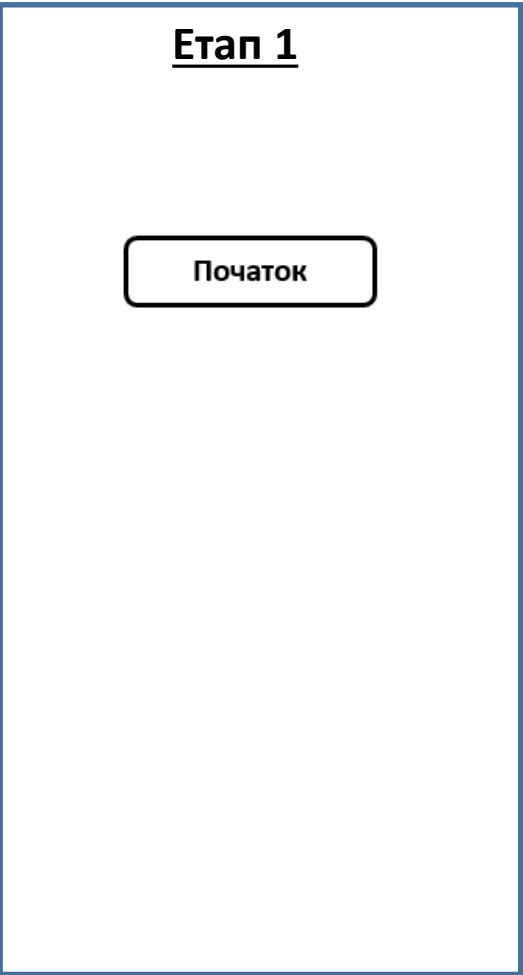
*Крок №0. Позначення величин*

Значення	Позначення	Тип
Вік користувача	<b>userAge</b>	Number (ціле)
Мінімальний допустимий вік	<b>minAllowedUserAge</b>	Number (ціле, константа = 18)

Приклад. На сайті банку задають питання стосовно віку користувача. Якщо вік <18, то повідомити про заборону доступу, інакше – привітати на сайті

*Крок №0. Позначення величин*

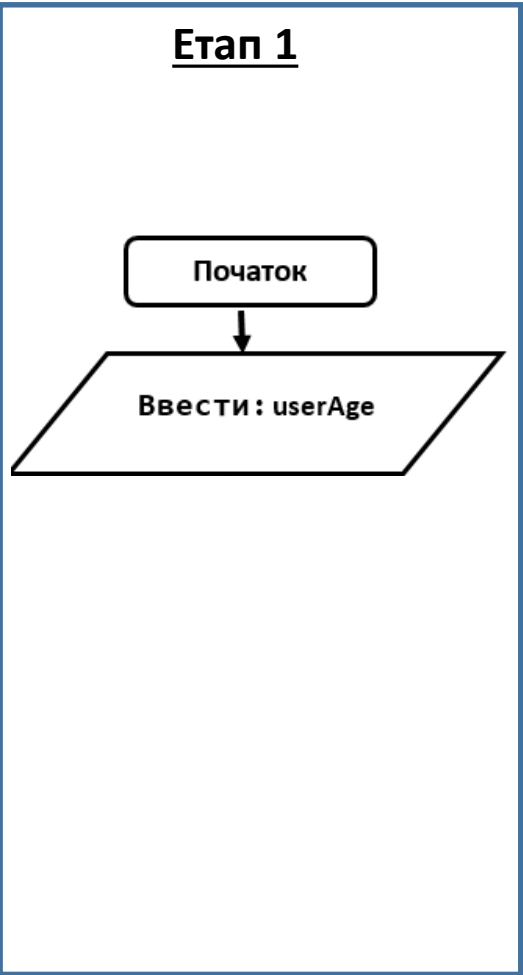
Значення	Позначення	Тип
Вік користувача	<b>userAge</b>	Number (ціле)
Мінімальний допустимий вік	<b>minAllowedUserAge</b>	Number (ціле, константа = 18)



Приклад. На сайті банку задають питання стосовно віку користувача. Якщо вік <18, то повідомити про заборону доступу, інакше – привітати на сайті

Крок №0. Позначення величин

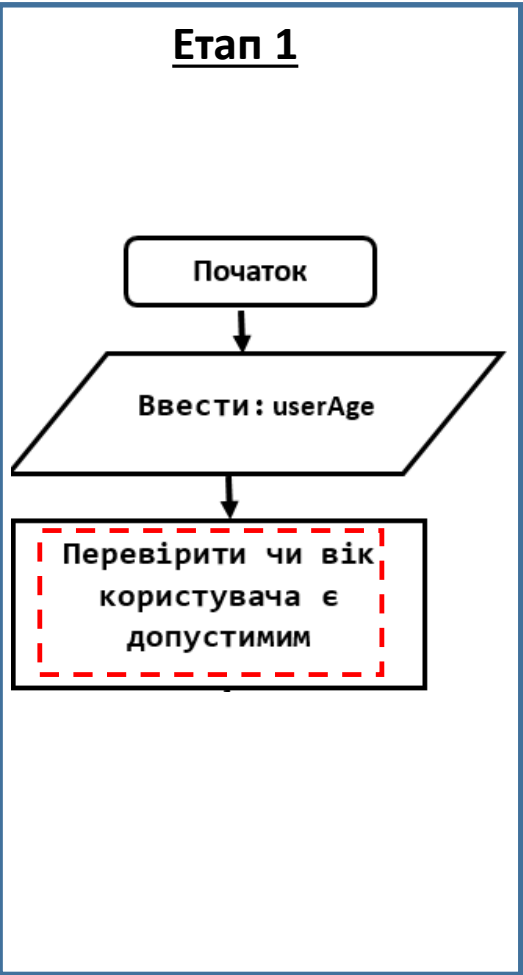
Значення	Позначення	Тип
Вік користувача	<b>userAge</b>	Number (ціле)
Мінімальний допустимий вік	<b>minAllowedUserAge</b>	Number (ціле, константа = 18)



Приклад. На сайті банку задають питання стосовно віку користувача. Якщо вік <18, то повідомити про заборону доступу, інакше – привітати на сайті

Крок №0. Позначення величин

Значення	Позначення	Тип
Вік користувача	<b>userAge</b>	Number (ціле)
Мінімальний допустимий вік	<b>minAllowedUserAge</b>	Number (ціле, константа = 18)

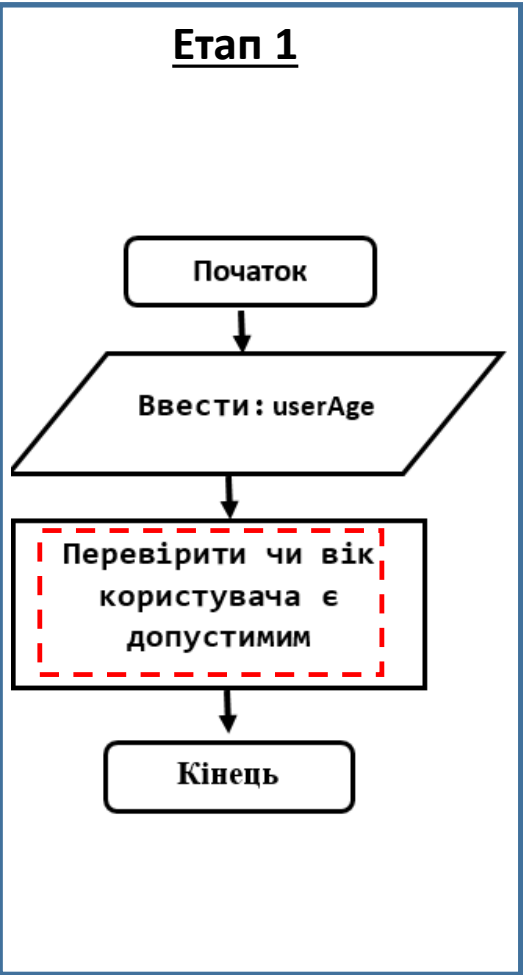




Приклад. На сайті банку задають питання стосовно віку користувача. Якщо вік <18, то повідомити про заборону доступу, інакше – привітати на сайті

Крок №0. Позначення величин

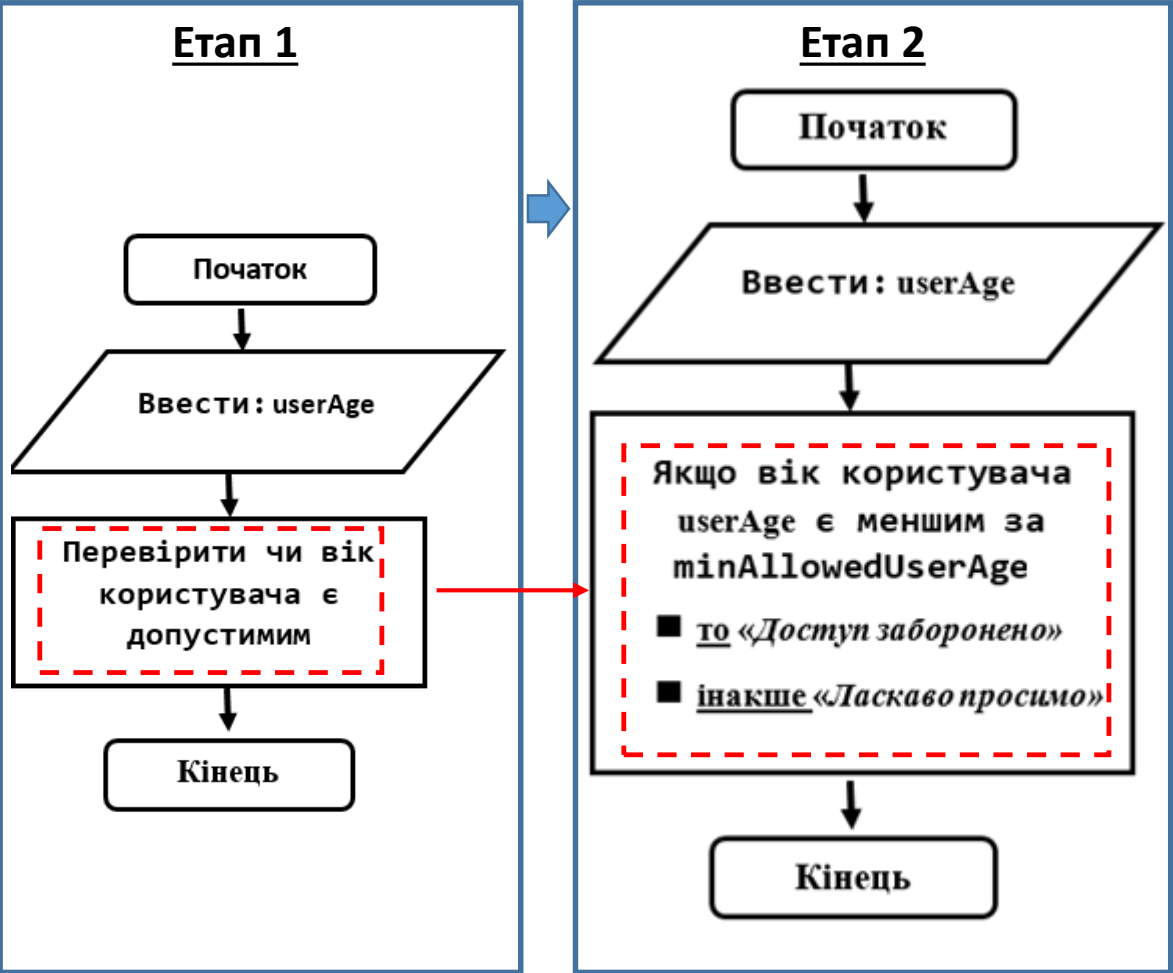
Значення	Позначення	Тип
Вік користувача	<code>userAge</code>	Number (ціле)
Мінімальний допустимий вік	<code>minAllowedUserAge</code>	Number (ціле, константа = 18)



Приклад. На сайті банку задають питання стосовно віку користувача. Якщо вік <18, то повідомити про заборону доступу, інакше – привітати на сайті

Крок №0. Позначення величин

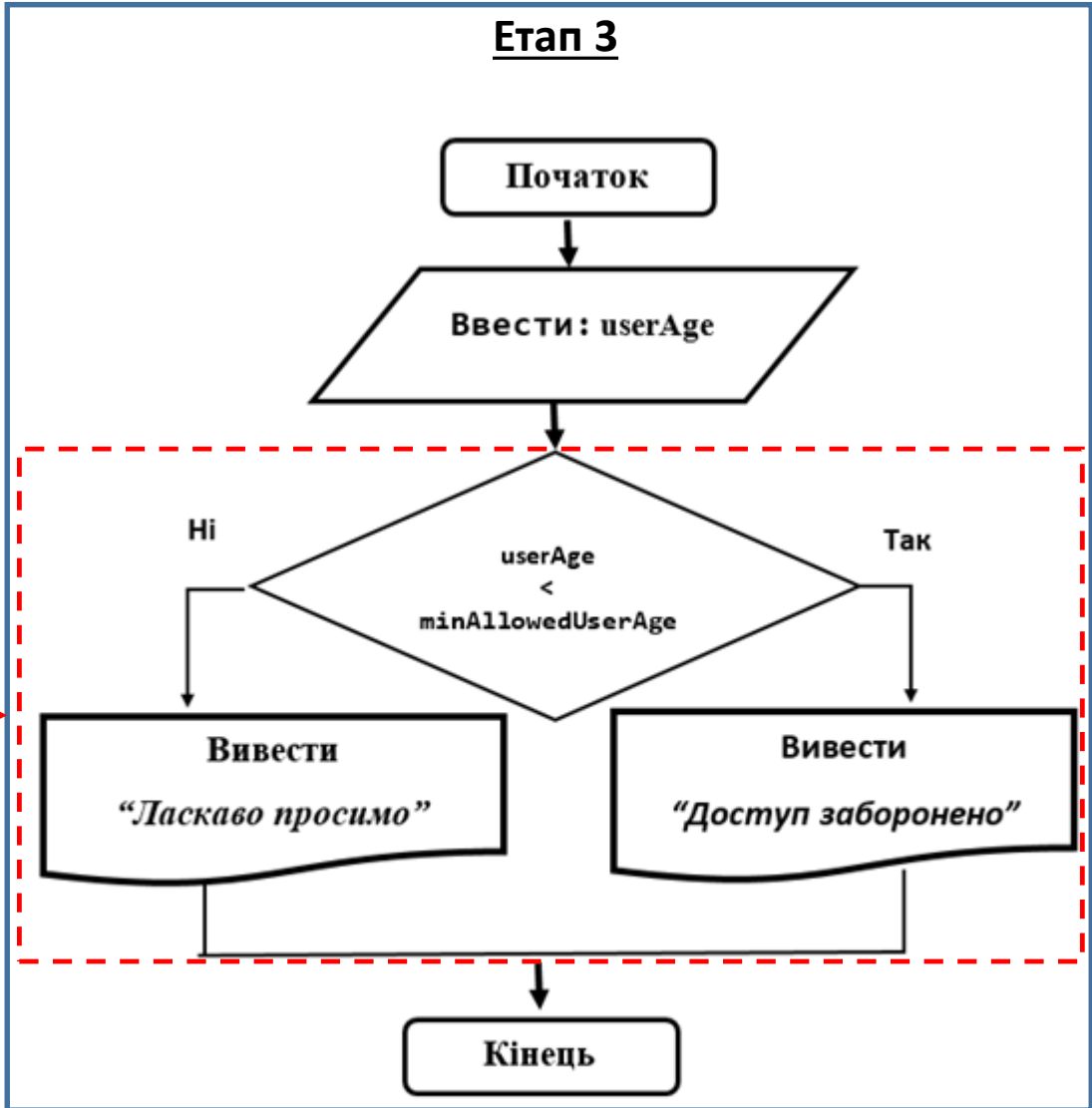
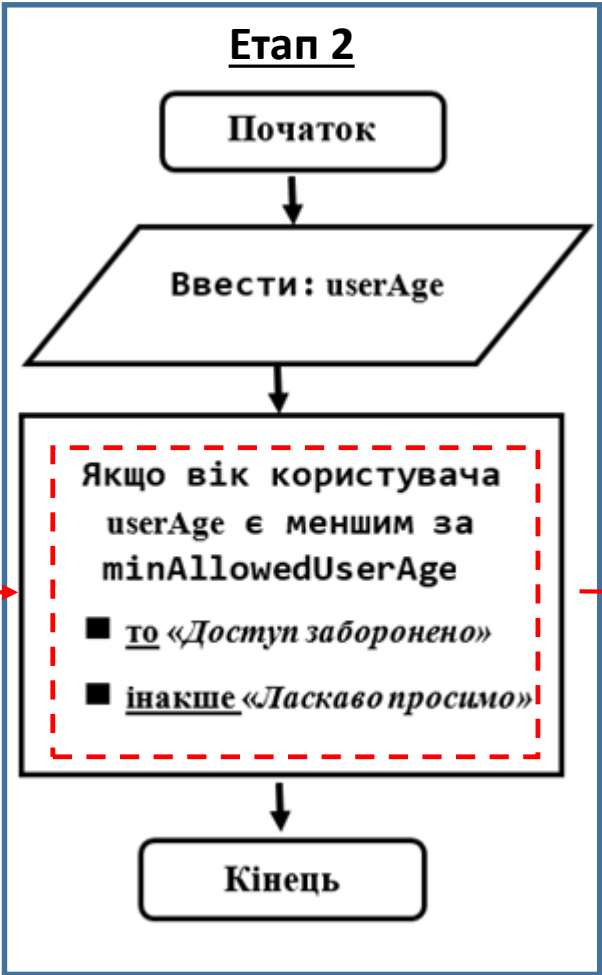
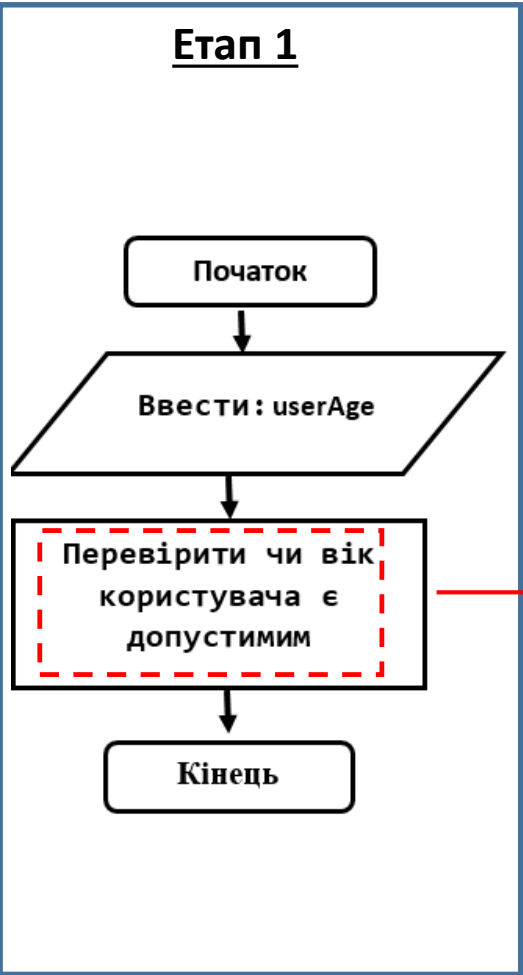
Значення	Позначення	Тип
Вік користувача	<code>userAge</code>	Number (ціле)
Мінімальний допустимий вік	<code>minAllowedUserAge</code>	Number (ціле, константа = 18)



Приклад. На сайті банку задають питання стосовно віку користувача. Якщо вік <18, то повідомити про заборону доступу, інакше – привітати на сайті

Крок №0. Позначення величин

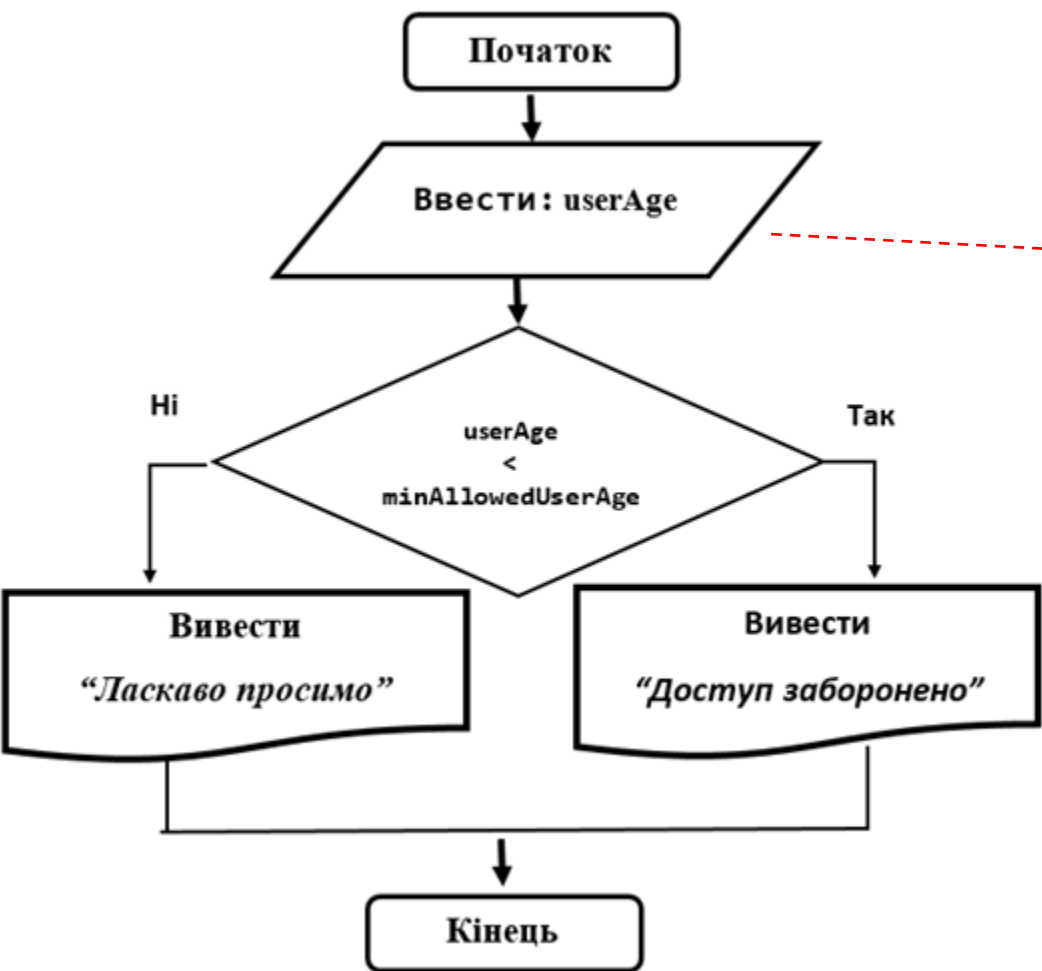
Значення	Позначення	Тип
Вік користувача	userAge	Number (ціле)
Мінімальний допустимий вік	minAllowedUserAge	Number (ціле, константа = 18)



Приклад. На сайті банку задають питання стосовно віку користувача. Якщо вік <18, то повідомити про заборону доступу, інакше – привітати на сайті

Крок №0. Позначення величин

Значення	Позначення	Тип
Вік користувача	<code>userAge</code>	Number (ціле)
Мінімальний допустимий вік	<code>minAllowedUserAge</code>	Number (ціле, константа = 18)

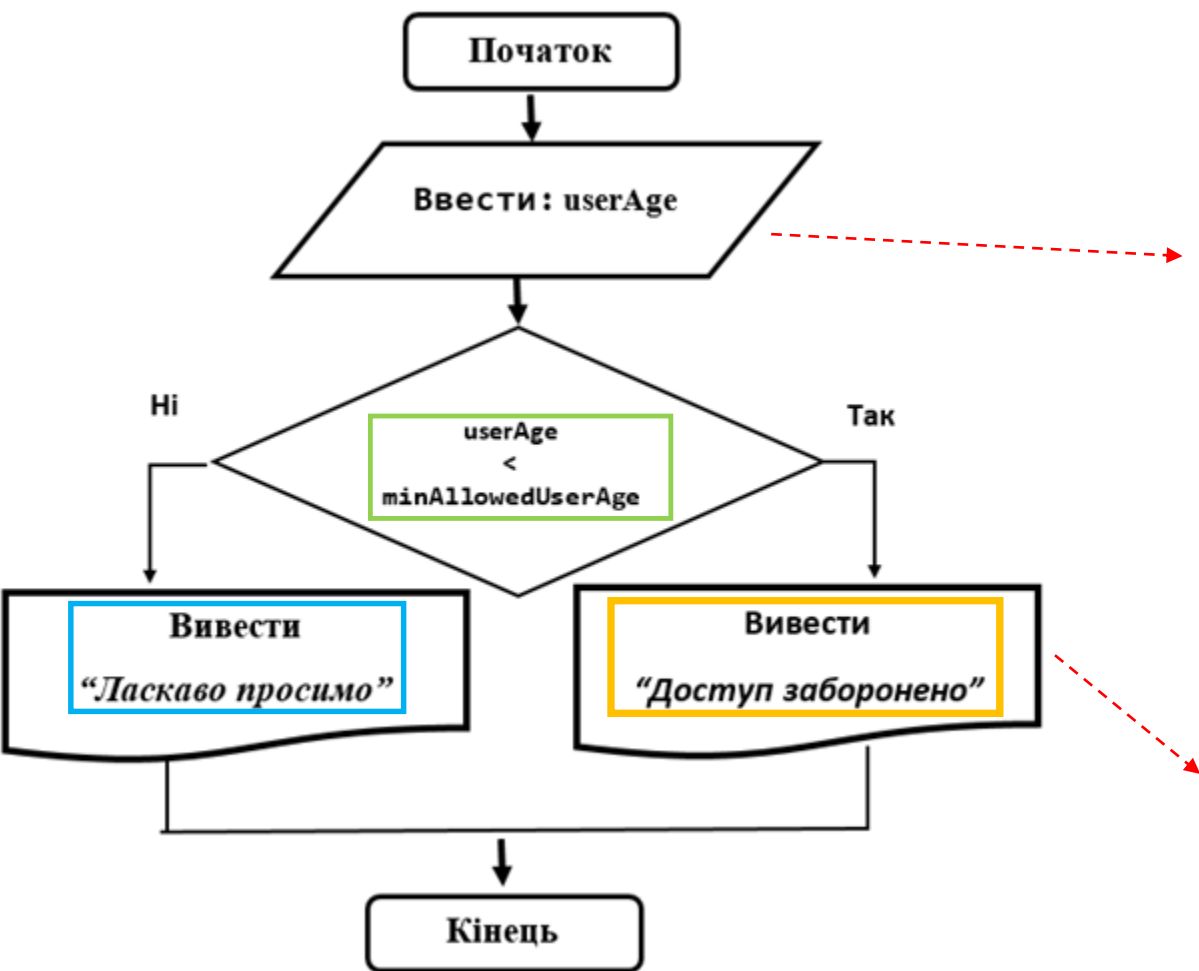


```
let userAge = parseInt(prompt('Введіть ваш вік', '18'))
const minAllowedUserAge = 18
```

Приклад. На сайті банку задають питання стосовно віку користувача. Якщо вік <18, то повідомити про заборону доступу, інакше – привітати на сайті

Крок №0. Позначення величин

Значення	Позначення	Тип
Вік користувача	userAge	Number (ціле)
Мінімальний допустимий вік	minAllowedUserAge	Number (ціле, константа = 18)



```
let userAge = parseInt(prompt('Введіть ваш вік', '18'))
const minAllowedUserAge = 18
```

Якщо (userAge < minAllowedUserAge)

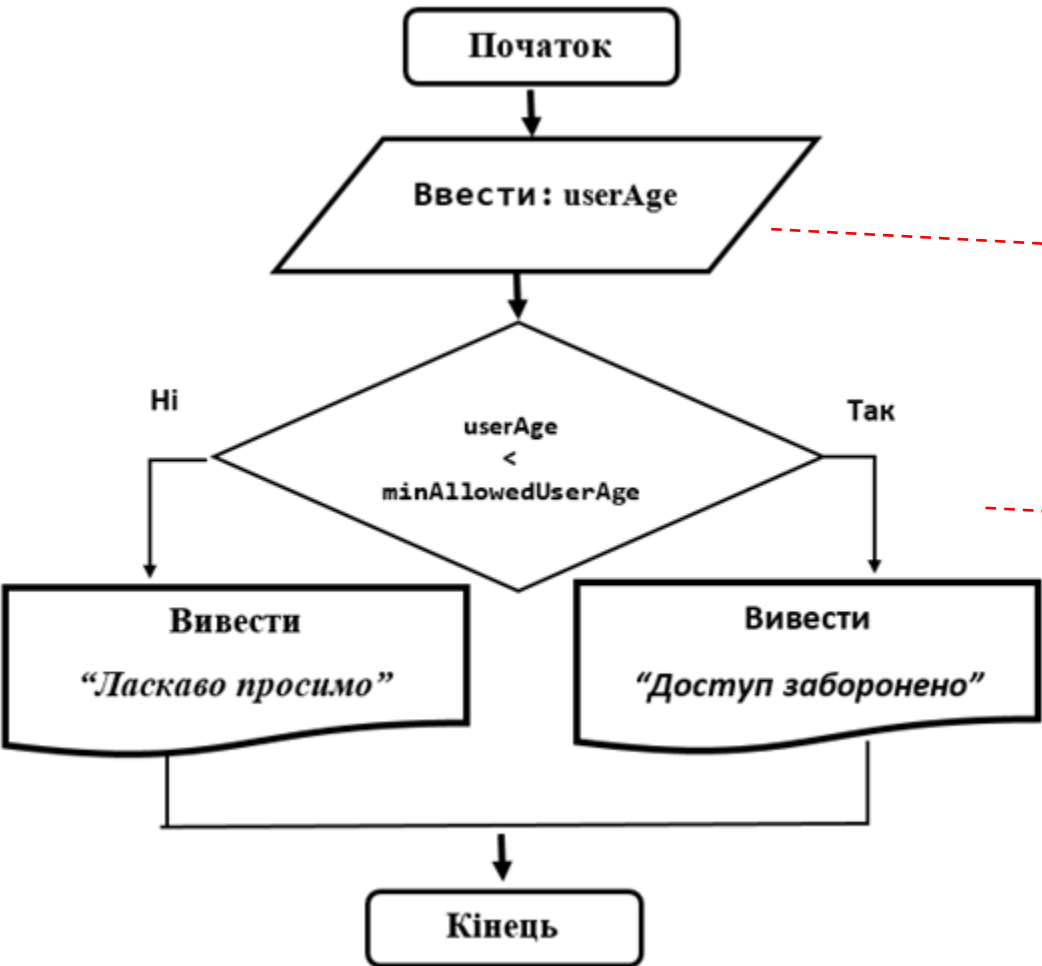
То (вивести «Ласкаво просимо»)

Інакше (вивести «Доступ заборонено»)

Приклад. На сайті банку задають питання стосовно віку користувача. Якщо вік <18, то повідомити про заборону доступу, інакше – привітати на сайті

Крок №0. Позначення величин

Значення	Позначення	Тип
Вік користувача	userAge	Number (ціле)
Мінімальний допустимий вік	minAllowedUserAge	Number (ціле, константа = 18)



```
let userAge = parseInt(prompt('Введіть ваш вік', '18'))
const minAllowedUserAge = 18

if (userAge < minAllowedUserAge) {
  console.log('Доступ заборонено')
} else {
  console.log('Ласкаво просимо')
}
```

Якщо (userAge < minAllowedUserAge)

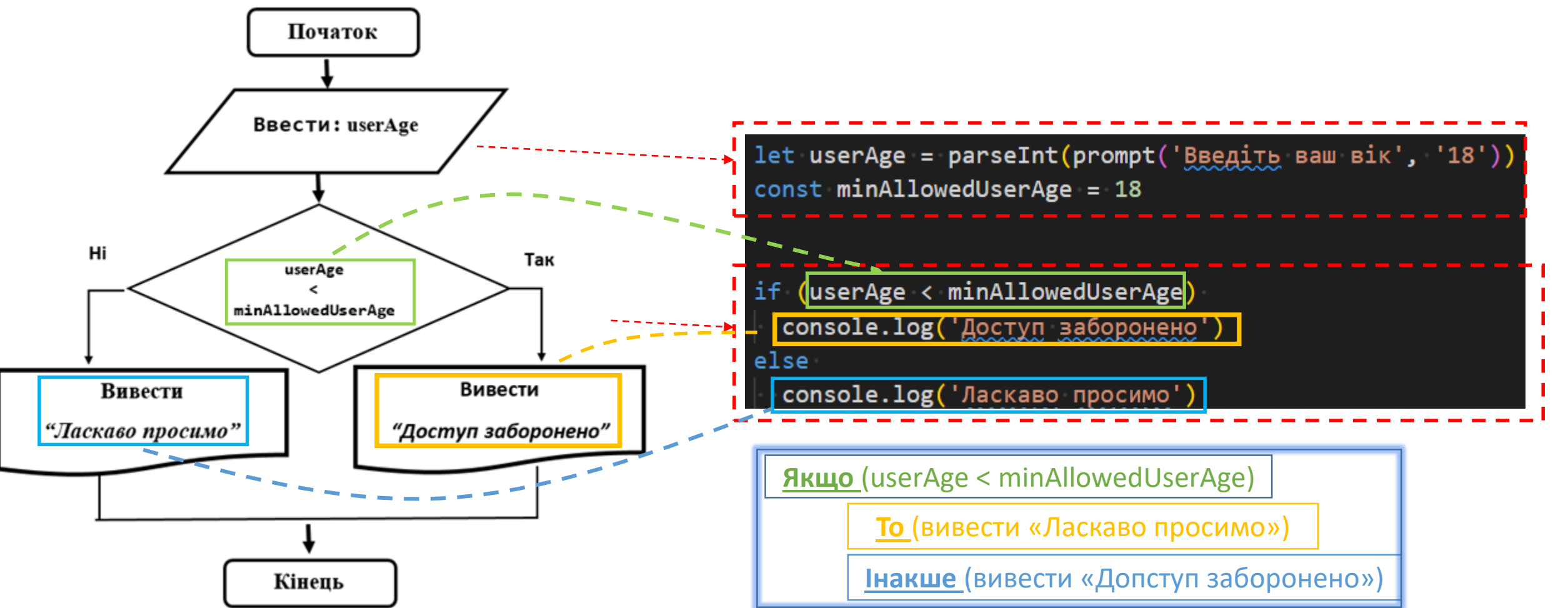
То (вивести «Ласкаво просимо»)

Інакше (вивести «Доступ заборонено»)

Приклад. На сайті банку задають питання стосовно віку користувача. Якщо вік <18, то повідомити про заборону доступу, інакше – привітати на сайті

Крок №0. Позначення величин

Значення	Позначення	Тип
Вік користувача	userAge	Number (ціле)
Мінімальний допустимий вік	minAllowedUserAge	Number (ціле, константа = 18)



Приклад. На сайті банку задають питання стосовно віку користувача. Якщо вік <18, то повідомити про заборону доступу, інакше – привітати на сайті

Крок №0. Позначення величин

Значення	Позначення	Тип
Вік користувача	<code>userAge</code>	Number (ціле)
Мінімальний допустимий вік	<code>minAllowedUserAge</code>	Number (ціле, константа = 18)

Якщо (`userAge < minAllowedUserAge`)

То (вивести «Ласкаво просимо»)

Інакше (вивести «Допступ заборонено»)

```
let userAge = parseInt(prompt('Введіть ваш вік', '18'))
const minAllowedUserAge = 18

if (userAge < minAllowedUserAge)
  console.log('Доступ заборонено')
else
  console.log('Ласкаво просимо')
```



Найбільше серед двох різних замінити на 0.

Найбільше серед двох різних замінити на 0.

Крок 0. Позначення величин

Величина	Позначення
Перше число	
Друге число	

Найбільше серед двох різних замінити на 0.

Крок 0. Позначення величин

Величина	Позначення
Перше число	firstNumber
Друге число	secondNumber

Найбільше серед двох різних замінити на 0.

Крок 0. Позначення величин

Величина	Позначення
Перше число	firstNumber
Друге число	secondNumber

Початок

Найбільше серед двох різних замінити на 0.

Крок 0. Позначення величин

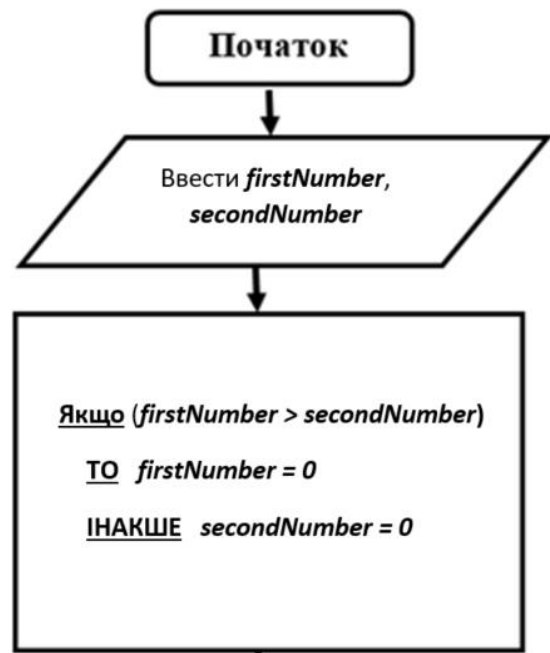
Величина	Позначення
Перше число	firstNumber
Друге число	secondNumber



Найбільше серед двох різних замінити на 0.

Крок 0. Позначення величин

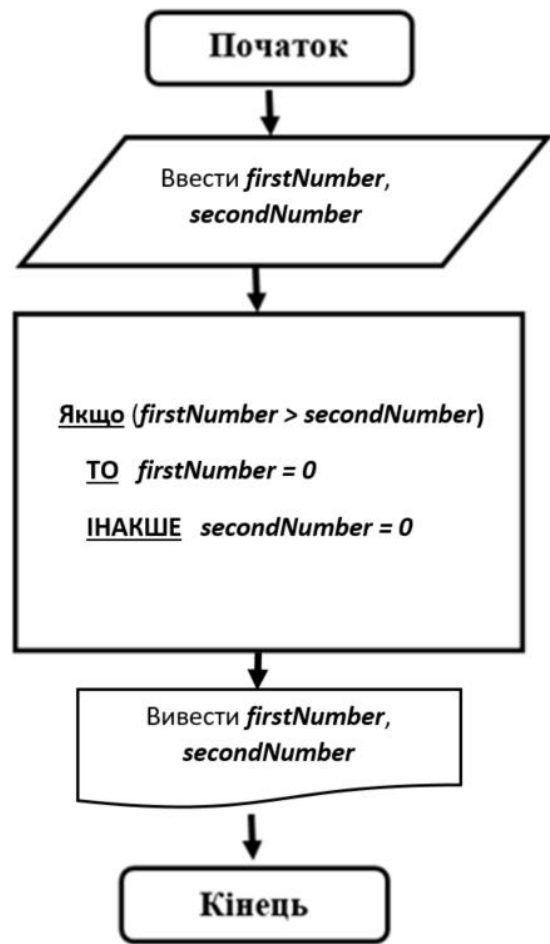
Величина	Позначення
Перше число	firstNumber
Друге число	secondNumber



Найбільше серед двох різних замінити на 0.

Крок 0. Позначення величин

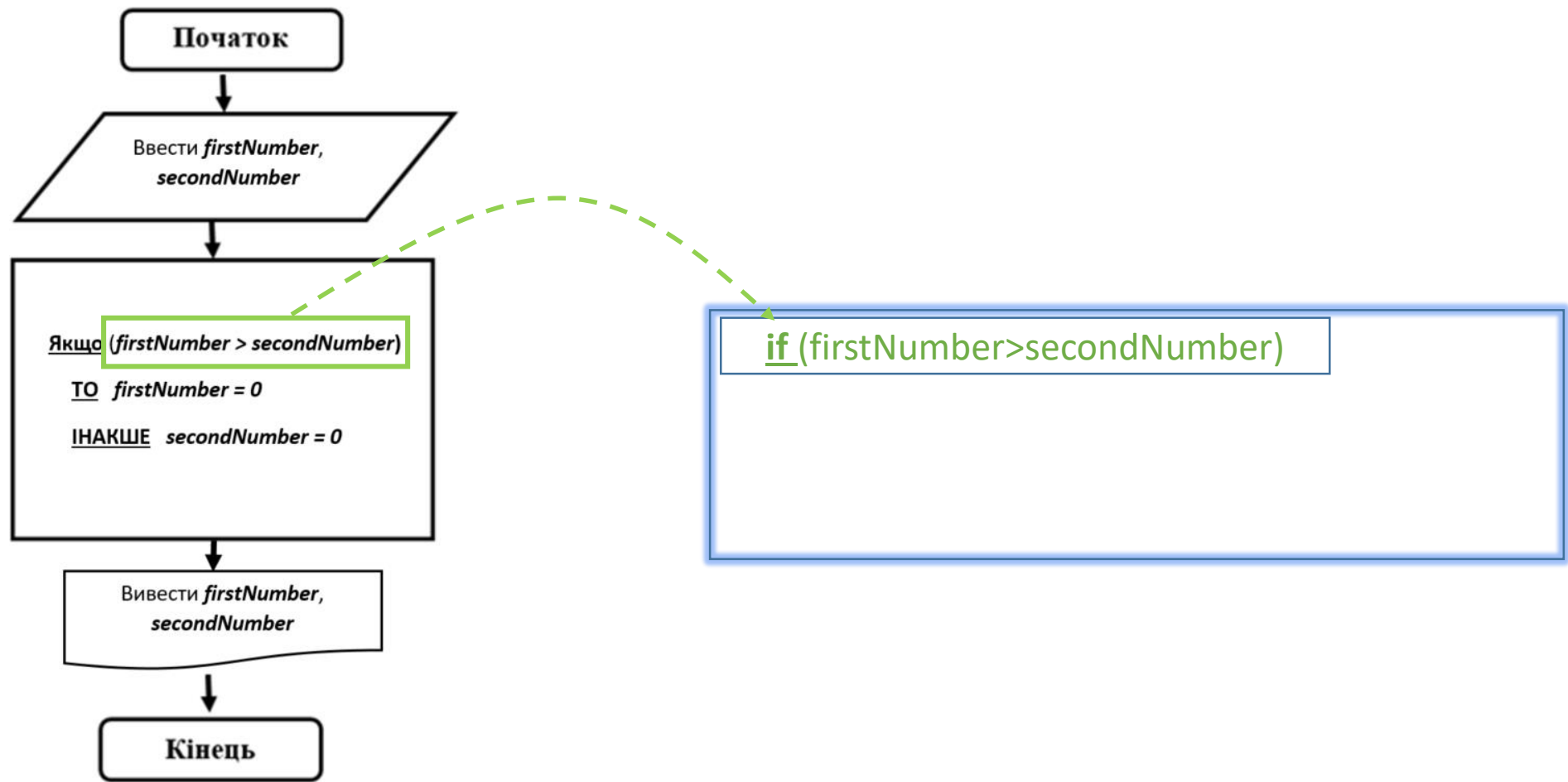
Величина	Позначення
Перше число	<code>firstNumber</code>
Друге число	<code>secondNumber</code>



Найбільше серед двох різних замінити на 0.

Крок 0. Позначення величин

Величина	Позначення
Перше число	firstNumber
Друге число	secondNumber

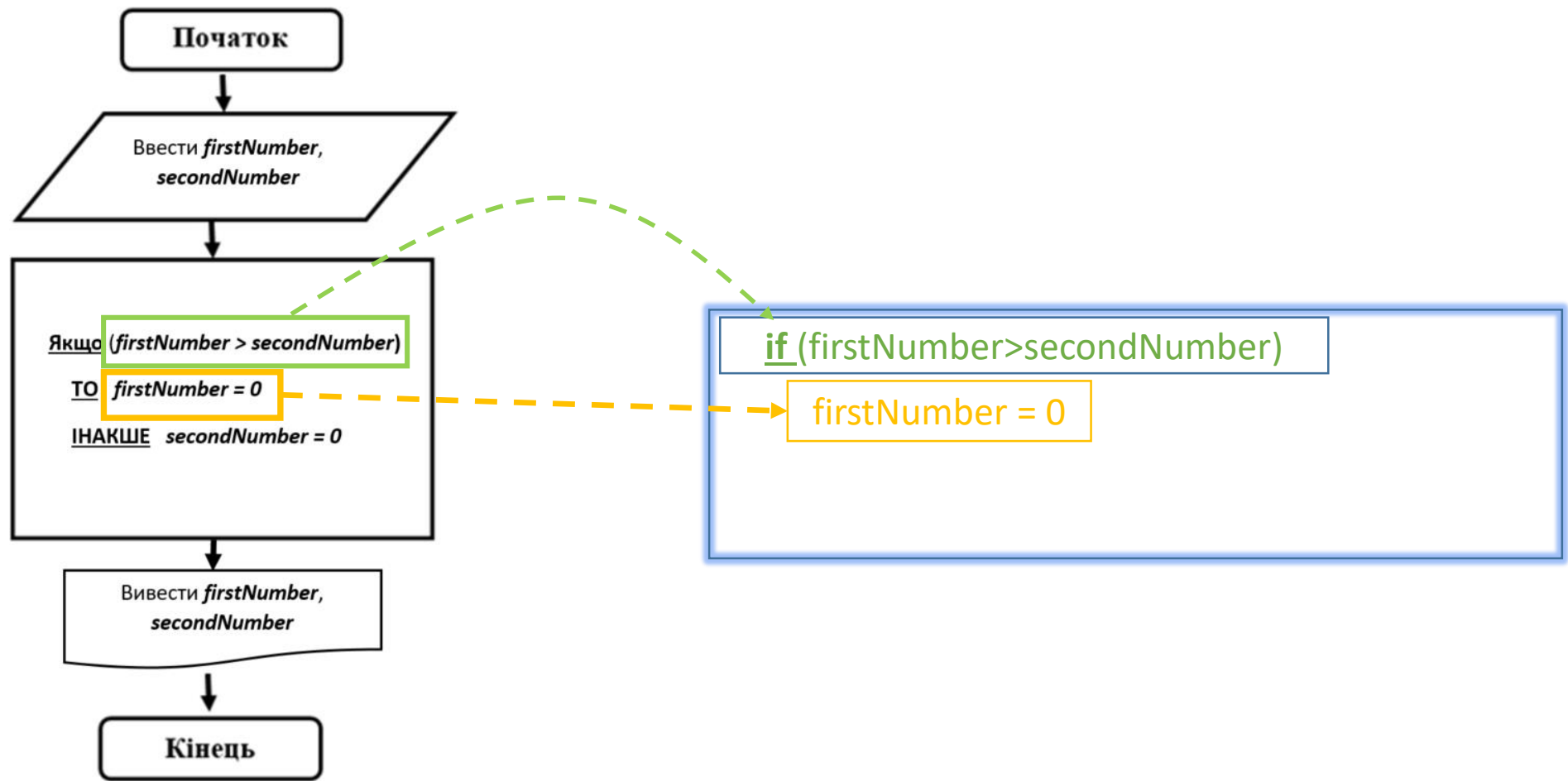




Найбільше серед двох різних замінити на 0.

Крок 0. Позначення величин

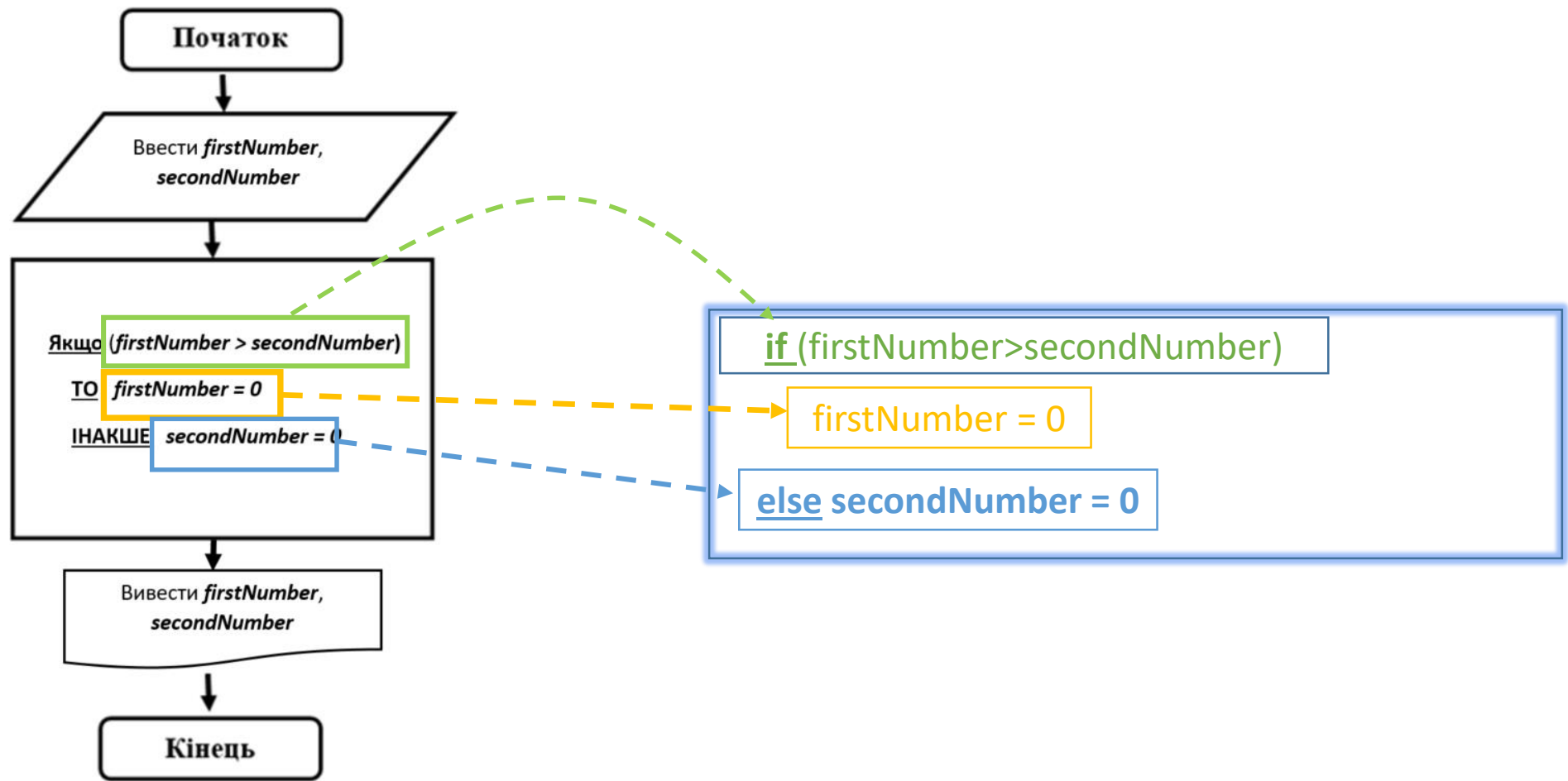
Величина	Позначення
Перше число	firstNumber
Друге число	secondNumber



Найбільше серед двох різних замінити на 0.

Крок 0. Позначення величин

Величина	Позначення
Перше число	firstNumber
Друге число	secondNumber



На роботу компанії приймає працівників від 32 до 45 років. З клавіатури вводиться вік претендента. З'ясувати, чи може він бути прийнятим на роботу.

Якщо при виконанні чи невиконанні деяких умов треба виконати більше ніж один оператор, то потрібно використовувати **складений оператор (блок)**, який записується за допомогою фігурних дужок

```
{  
    оператор1;  
    оператор2;  
    . . . . .  
}
```

Програмна структура	Аналог на мові блок-схем	Приклад
<pre>if (&lt;умова&gt;) {     &lt;оператор1.1&gt;     . . . . .     &lt;оператор1.N&gt; } else {     &lt;оператор2.1&gt;     . . . . .     &lt;оператор2.M&gt; }</pre>	<pre>graph TD     Entry(( )) --&gt; Uмова{умова}     Uмова -- - --&gt; Op2[Оператор2.1 ... Оператор2.M]     Uмова -- + --&gt; Op1[Оператор1.1 ... Оператор1.N]     Op2 --&gt; Exit(( ))     Op1 --&gt; Exit     Exit --&gt; ExitArrow[ ]</pre>	<pre>if(&lt;x&gt;y) {     max=x     min=y } else {     max=y     min=x }</pre>

Якщо при виконанні чи невиконанні деяких умов треба виконати більше ніж один оператор, то потрібно використовувати **складений оператор (блок)**, який записується за допомогою фігурних дужок

```
{
    оператор1;
    оператор2;
    . . . . .
}
```

Програмна структура	Аналог на мові блок-схем	Приклад
<pre>if (&lt;умова&gt;) {     &lt;оператор1.1&gt;     . . . . .     &lt;оператор1.N&gt; } else {     &lt;оператор2.1&gt;     . . . . .     &lt;оператор2.M&gt; }</pre>		<pre>if(&lt;х&gt;&lt;у&gt;) {     max=x     min=y } else {     max=y     min=x }</pre>

Скорочена форма

Програмна структура	Аналог на мові блок-схем	Приклад
<pre>if (&lt;умова&gt;) {     &lt;оператор1&gt;     . . . . .     &lt;операторN&gt; }</pre>		<pre>if(&lt;х&gt;&lt;0&gt;) {     z=1/х     l=y/х }</pre>

# Тернарний оператор

Іноді буває ситуація, коли значення виразу повинно дорівнювати одному із двох значень у залежності від виконання чи невиконання деякої умови. Наприклад, потрібно залежно від умови присвоїти змінній певне значення. В цьому випадку можна використати тернарний оператор.

Загальна форма	Змінна = <span>умова</span> ? <span>значення1 (умова викон.)</span> : <span>значення2 (умова не викон.)</span> ;
----------------	------------------------------------------------------------------------------------------------------------------

# Тернарний оператор

Іноді буває ситуація, коли значення виразу повинно дорівнювати одному із двох значень у залежності від виконання чи невиконання деякої умови. Наприклад, потрібно залежно від умови присвоїти змінній певне значення. В цьому випадку можна використати тернарний оператор.

Загальна  
форма

Змінна = умова ? значення1 (умова викон.) : значення2 (умова не викон.) :

----- аналог з умовним оператором -----

if ( умова )

    змінна = значення1

else

    змінна = значення2



# Тернарний оператор

Іноді буває ситуація, коли значення виразу повинно дорівнювати одному із двох значень у залежності від виконання чи невиконання деякої умови. Наприклад, потрібно залежно від умови присвоїти змінній певне значення. В цьому випадку можна використати тернарний оператор.

Загальна форма	<pre>Змінна = умова ? значення1 (умова викон.) : значення2 (умова не викон.);</pre> <p>----- аналог з умовним оператором -----</p> <pre>if ( умова )     змінна = значення1; else     змінна = значення2;</pre>
Приклад	<pre>max = a &gt; b ? a : b;</pre> <p>----- аналог з умовним оператором -----</p> <pre>if (a &gt; b)     max = a; else     max = b;</pre>



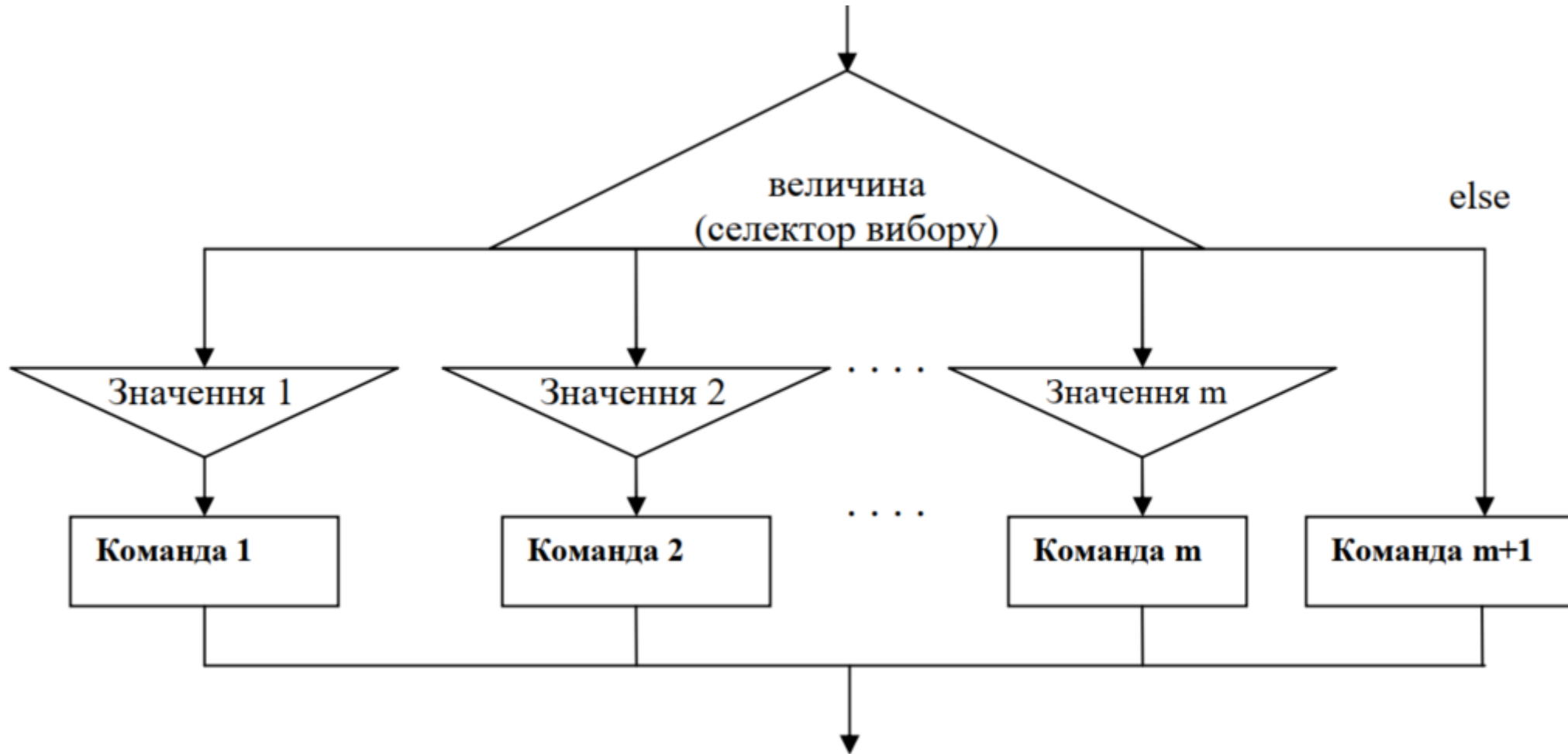
Приклад. Скласти скрипт для знаходження більшого з двох чисел.

```
let a=parseFloat(prompt('Введіть перше число',''))  
let b= parseFloat(prompt('Введіть друге число',''))  
let max=(a>b) ? a : b  
alert('Більше число рівне '+max)
```

З клавіатури вводиться кількість балів. Вивести на екран оцінку прописом (задовільно, добре чи відмінно).

# ОПЕРАТОР ВИБОРУ

Якщо у залежності від значення деякого виразу потрібно виконати оду із команд,  
то можна використати оператор вибору (щоб не писати багато умовних операторів)



**Приклад.** Користувач задає оцінку у п'ятибальній шкалі. Необхідно встановити, текстове представлення цієї оцінки (5 – «відмінно», 4 – «добре», 3 – «задовільно», 1 і 2 – «незадовільно»).

**Приклад.** Користувач задає оцінку у п'ятибальній шкалі. Необхідно встановити, текстове представлення цієї оцінки (5 – «відмінно», 4 – «добре», 3 – «задовільно», 1 і 2 – «незадовільно»).



Score

**Приклад.** Користувач задає оцінку у п'ятибальній шкалі. Необхідно встановити, текстове представлення цієї оцінки (5 – «відмінно», 4 – «добре», 3 – «задовільно», 1 і 2 – «незадовільно»).



Score

*Розв'язання*

При побудові алгоритму використаємо структуру вибору.

Початок

**Приклад.** Користувач задає оцінку у п'ятибальній шкалі. Необхідно встановити, текстове представлення цієї оцінки (5 – «відмінно», 4 – «добре», 3 – «задовільно», 1 і 2 – «незадовільно»).

Score

*Розв'язання*

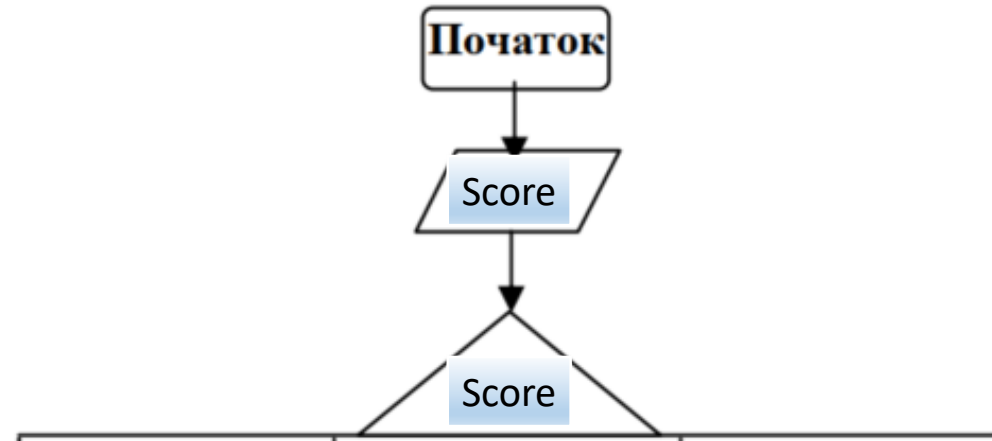
При побудові алгоритму використаємо структуру вибору.



**Приклад.** Користувач задає оцінку у п'ятибальній шкалі. Необхідно встановити, текстове представлення цієї оцінки (5 – «відмінно», 4 – «добре», 3 – «задовільно», 1 і 2 – «незадовільно»).

*Розв'язання*

При побудові алгоритму використаємо структуру вибору.

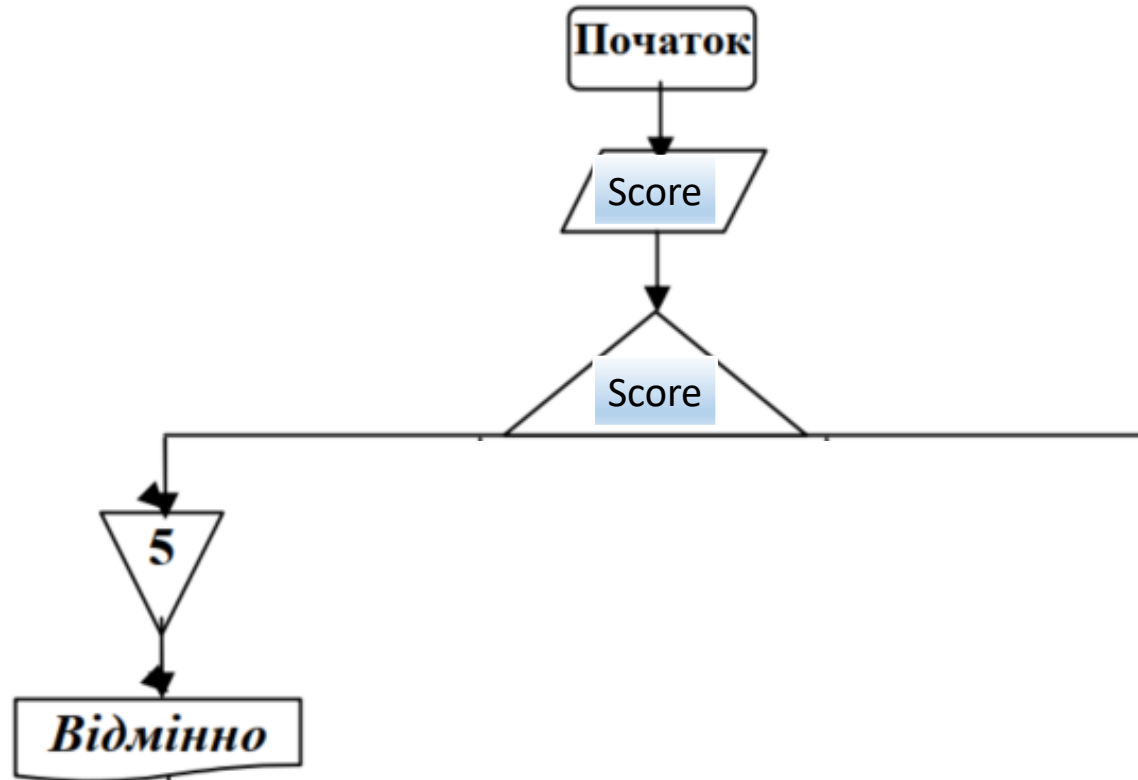




**Приклад.** Користувач задає оцінку у п'ятибальній шкалі. Необхідно встановити, текстове представлення цієї оцінки (5 – «відмінно», 4 – «добре», 3 – «задовільно», 1 і 2 – «незадовільно»).  
Score

*Розв'язання*

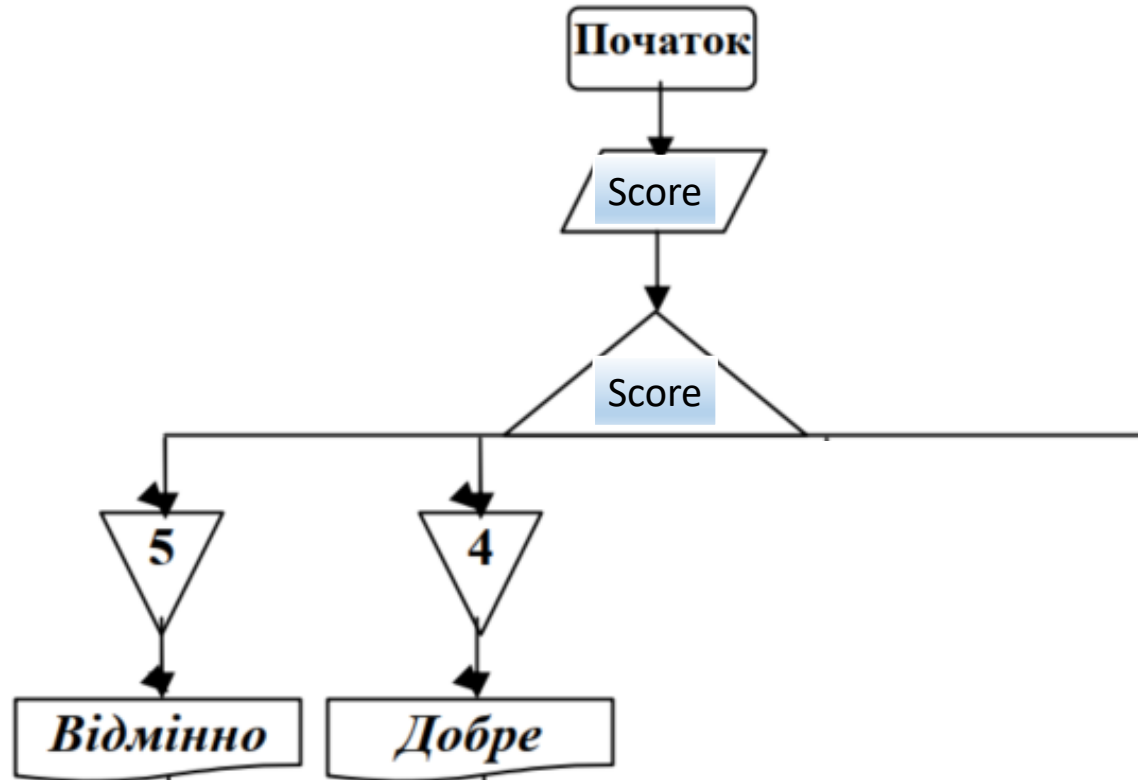
При побудові алгоритму використаємо структуру вибору.



**Приклад.** Користувач задає оцінку у п'ятибальній шкалі. Необхідно встановити, текстове представлення цієї оцінки (5 – «відмінно», 4 – «добре», 3 – «задовільно», 1 і 2 – «незадовільно»).

*Розв'язання*

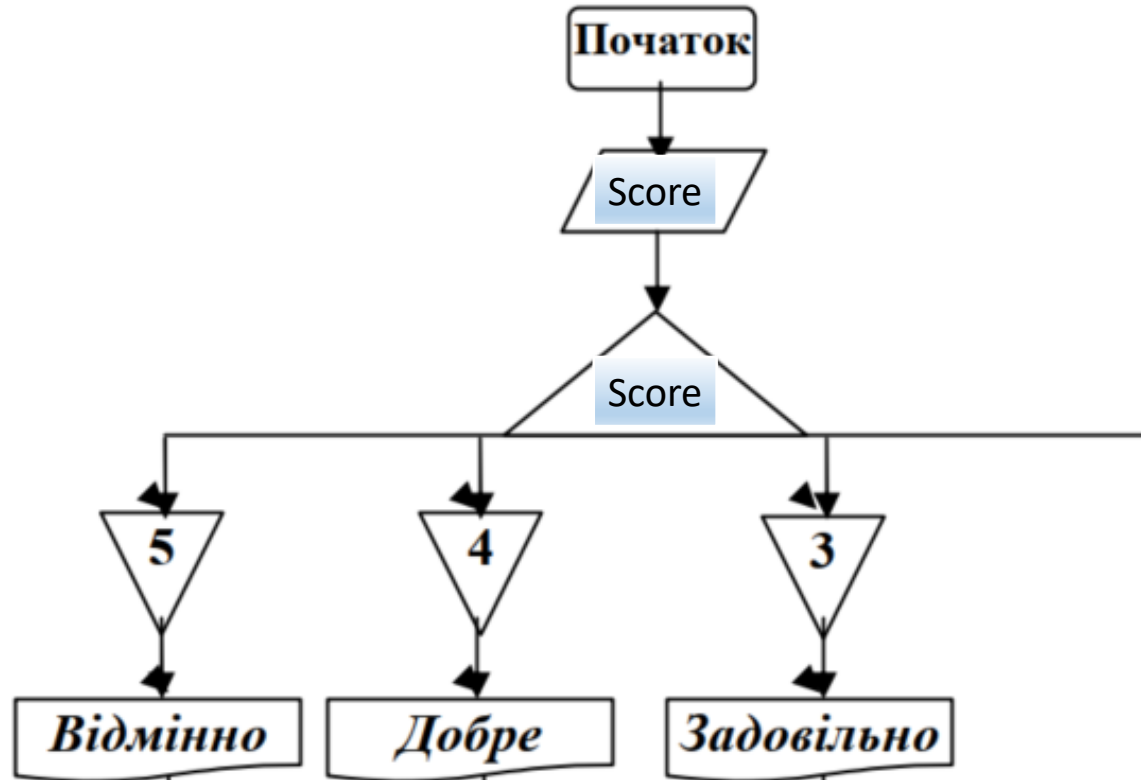
При побудові алгоритму використаємо структуру вибору.



**Приклад.** Користувач задає оцінку у п'ятибальній шкалі. Необхідно встановити, текстове представлення цієї оцінки (5 – «відмінно», 4 – «добре», 3 – «задовільно», 1 і 2 – «незадовільно»). → Score

*Розв'язання*

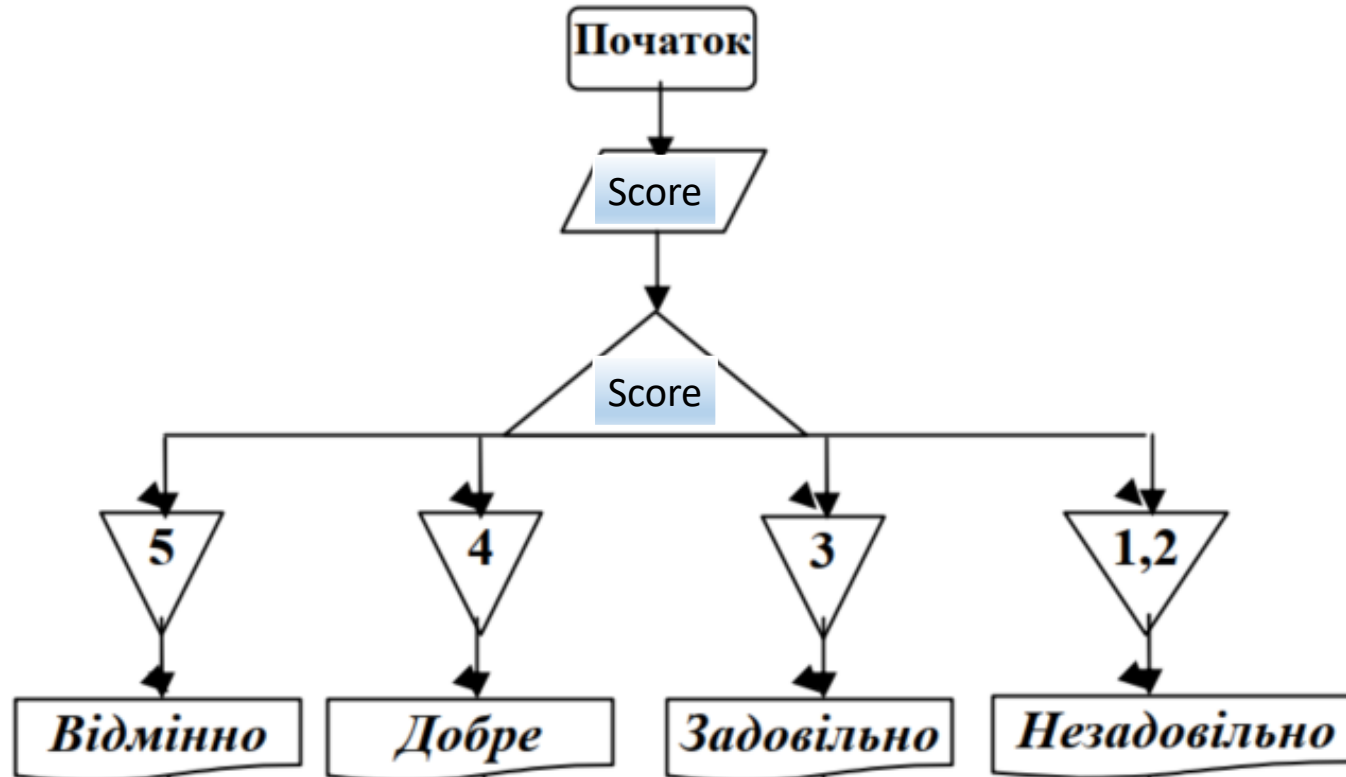
При побудові алгоритму використаємо структуру вибору.



**Приклад.** Користувач задає оцінку у п'ятибальній шкалі. Необхідно встановити, текстове представлення цієї оцінки (5 – «відмінно», 4 – «добре», 3 – «задовільно», 1 і 2 – «незадовільно»). Score

*Розв'язання*

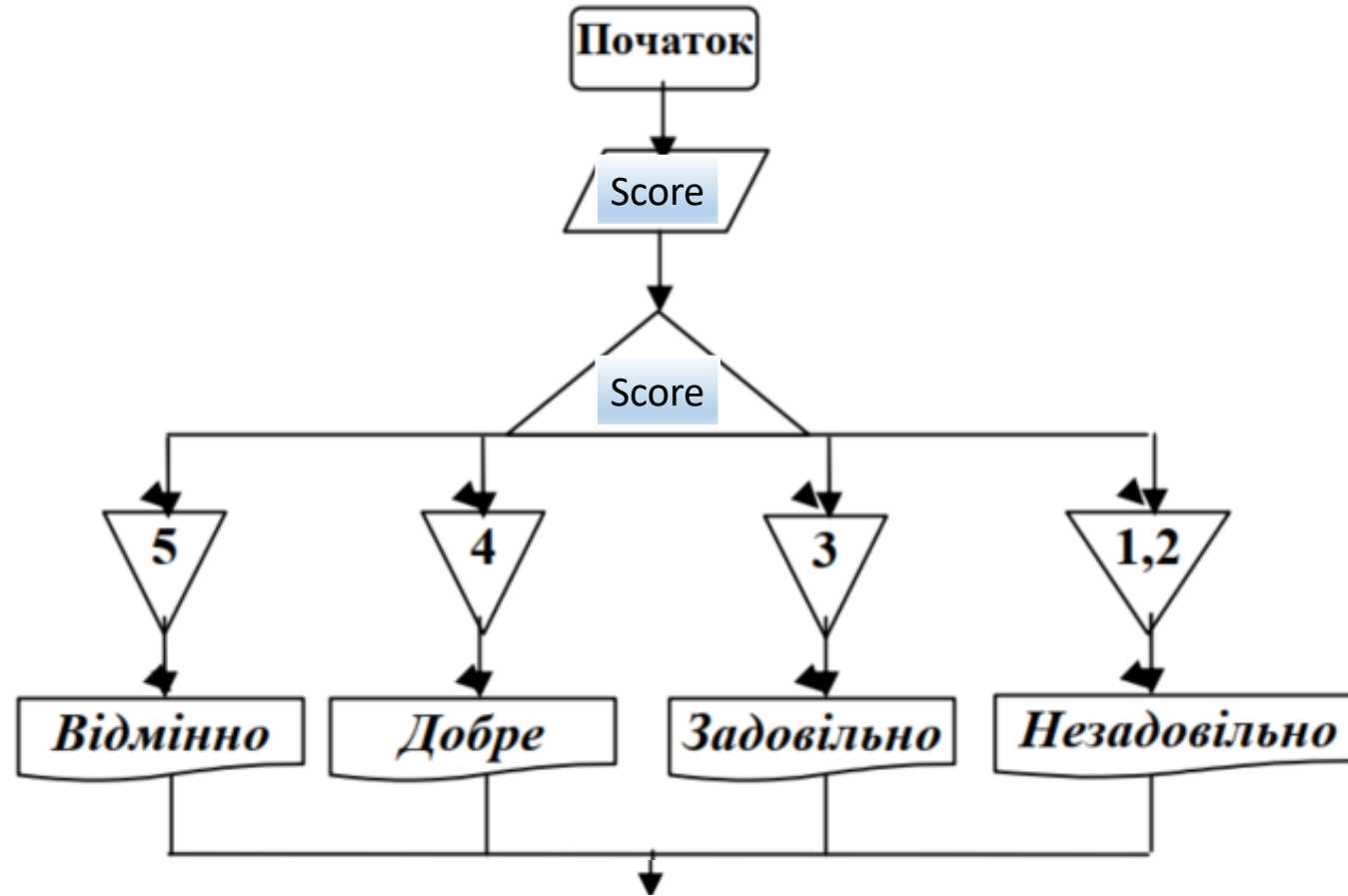
При побудові алгоритму використаємо структуру вибору.



**Приклад.** Користувач задає оцінку у п'ятибальній шкалі. Необхідно встановити, текстове представлення цієї оцінки (5 – «відмінно», 4 – «добре», 3 – «задовільно», 1 і 2 – «незадовільно»).  
Score

*Розв'язання*

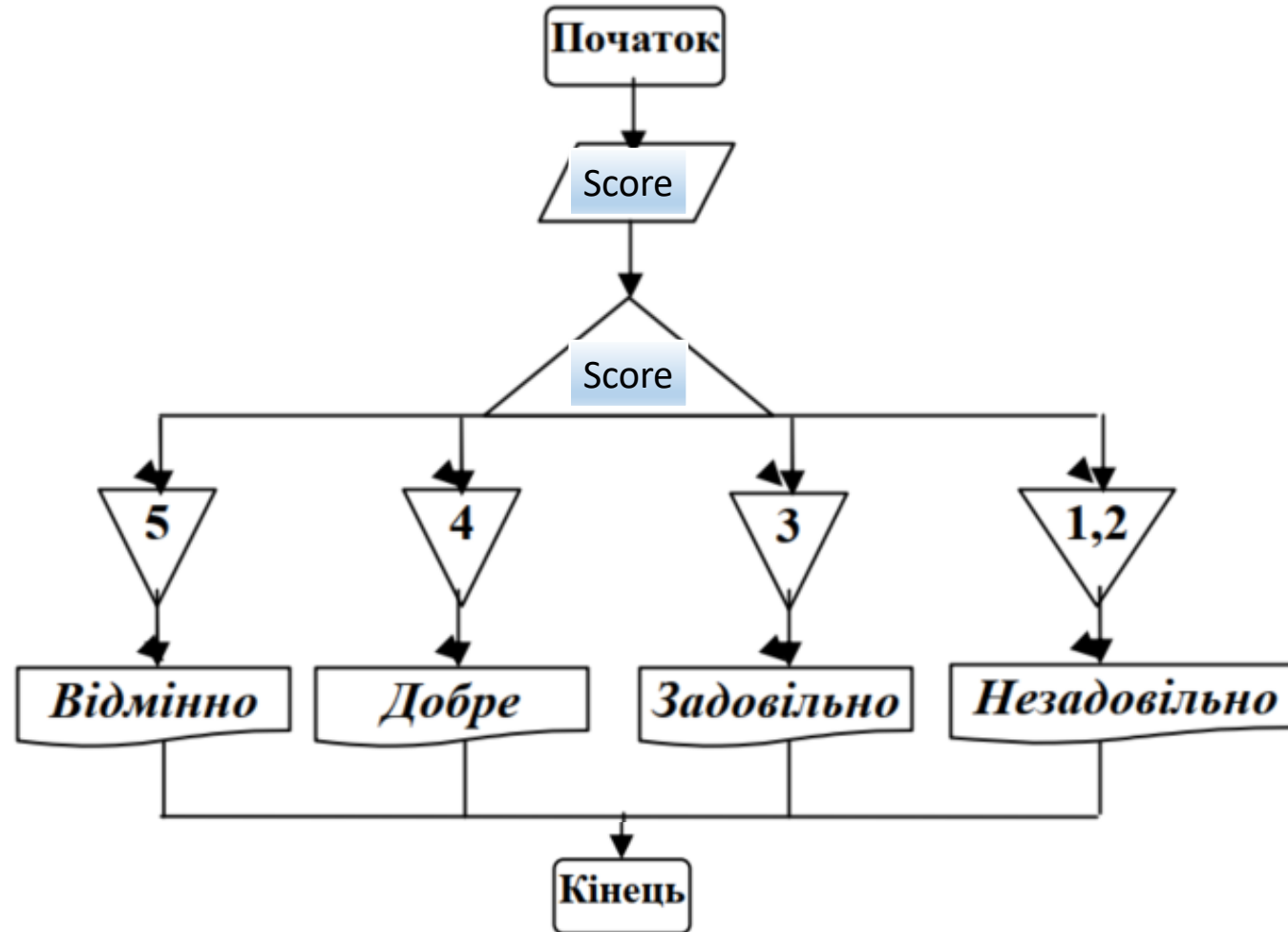
При побудові алгоритму використаємо структуру вибору.



**Приклад.** Користувач задає оцінку у п'ятибальній шкалі. Необхідно встановити, текстове представлення цієї оцінки (5 – «відмінно», 4 – «добре», 3 – «задовільно», 1 і 2 – «незадовільно»).  
Score

*Розв'язання*

При побудові алгоритму використаємо структуру вибору.

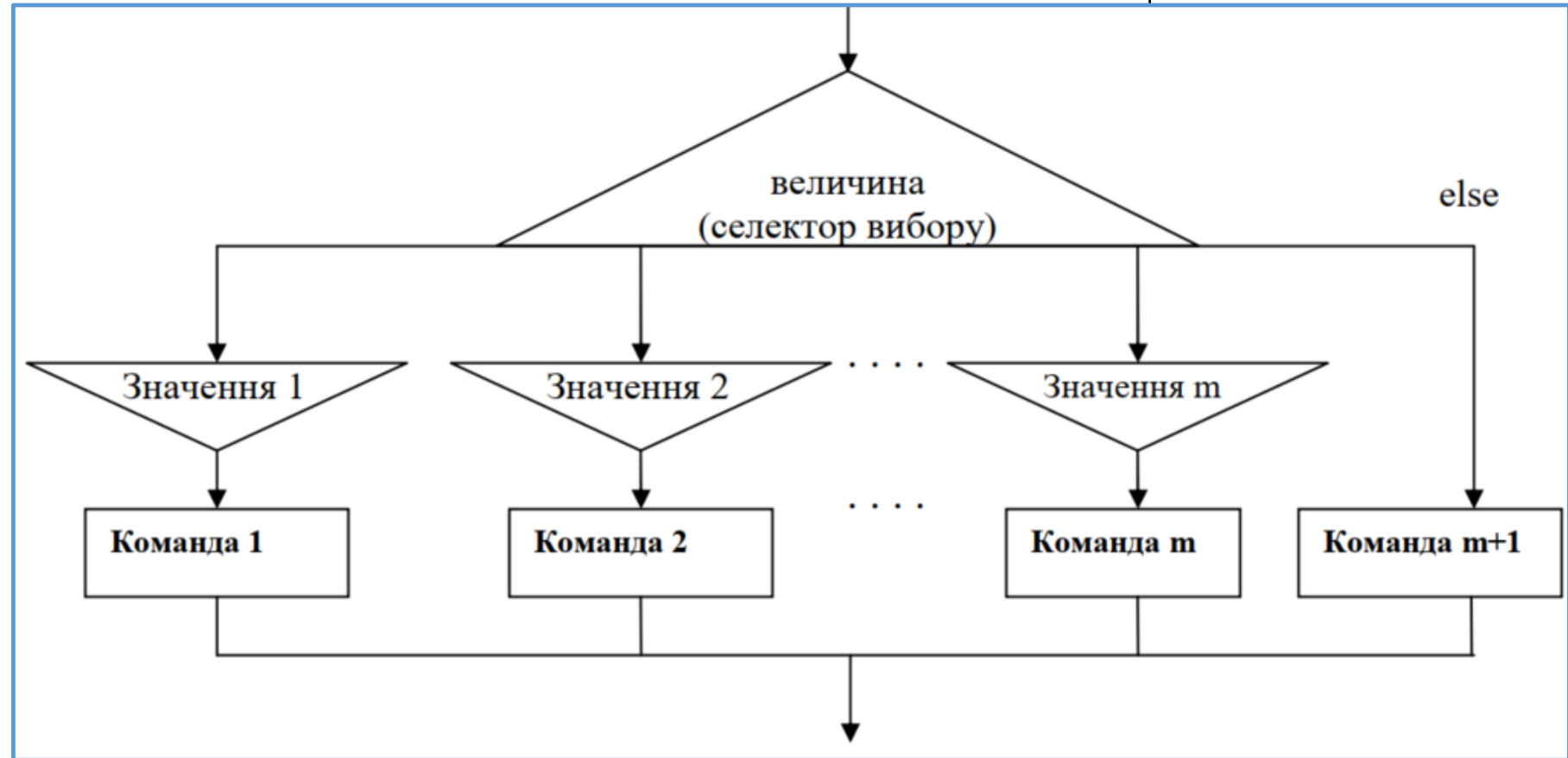


# ОПЕРАТОР ВИБОРУ SWITCH

Конструкція switch замінює собою відразу кілька if. Це оператор викорситовується тоді, коли у залежності від значення деякої величини (кількість можливих значень є невеликою) потрібно виконати ті чи інші оператори.

Це більш наочний спосіб порівняти вираз (в сенсі оператора «===») відразу з декількома варіантами.

```
switch (селектор) {  
    case значення 1 : команда 1  
        break;  
    case значення 2 : команда 2  
        break;  
    . . . . .  
    case значення m : команда m  
        break;  
  
    default:  
        команда m+1  
}
```

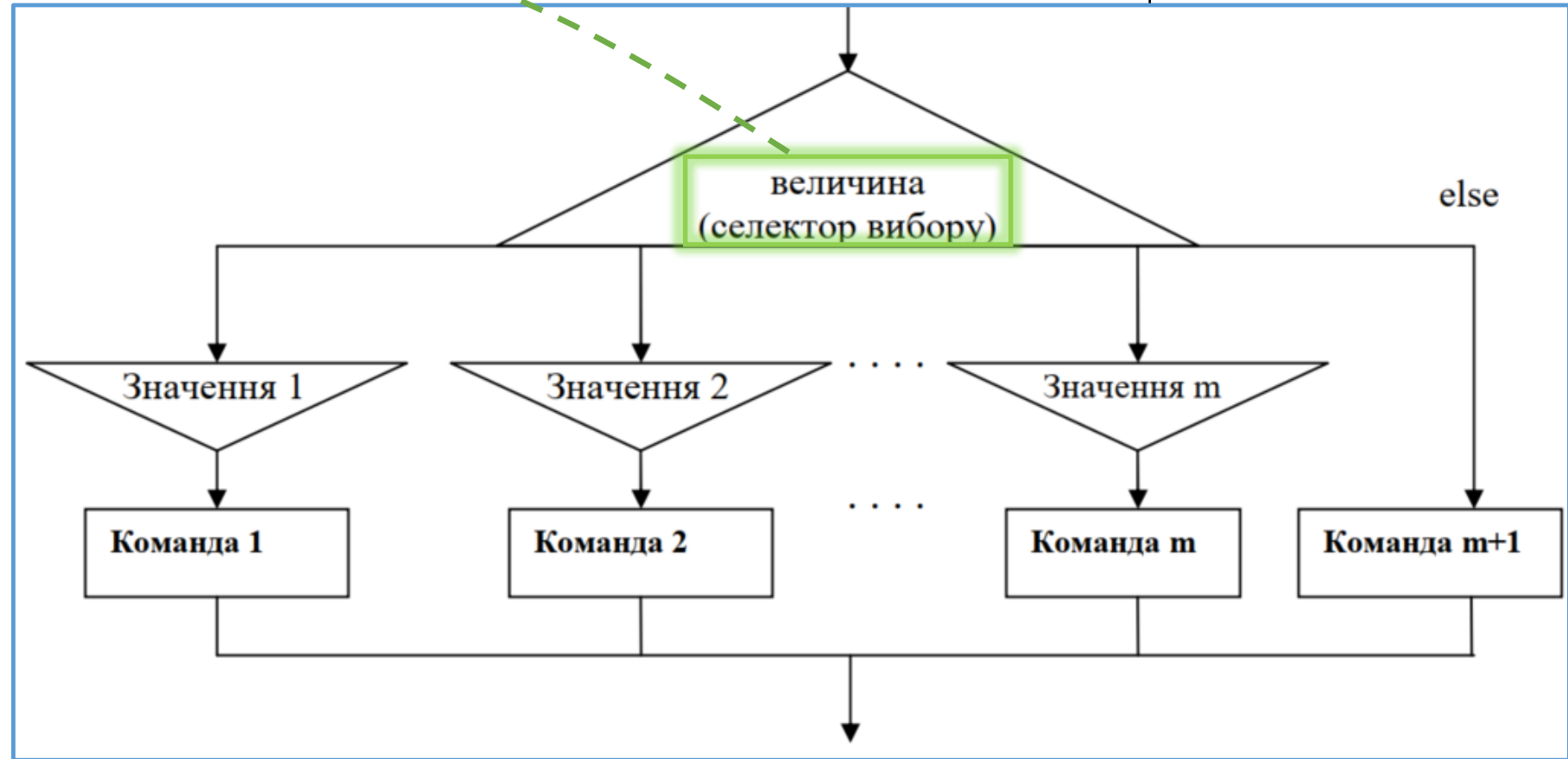


# ОПЕРАТОР ВИБОРУ SWITCH

Конструкція switch замінює собою відразу кілька if. Це оператор викорситовується тоді, коли у залежності від значення деякої величини (кількість можливих значень є невеликою) потрібно виконати ті чи інші оператори.

Це більш наочний спосіб порівняти вираз (в сенсі оператора «===») відразу з декількома варіантами.

```
switch (селектор) {
```



```
}
```



# ОПЕРАТОР ВИБОРУ SWITCH

Конструкція switch замінює собою відразу кілька if. Це оператор викорситовується тоді, коли у залежності від значення деякої величини (кількість можливих значень є невеликою) потрібно виконати ті чи інші оператори.

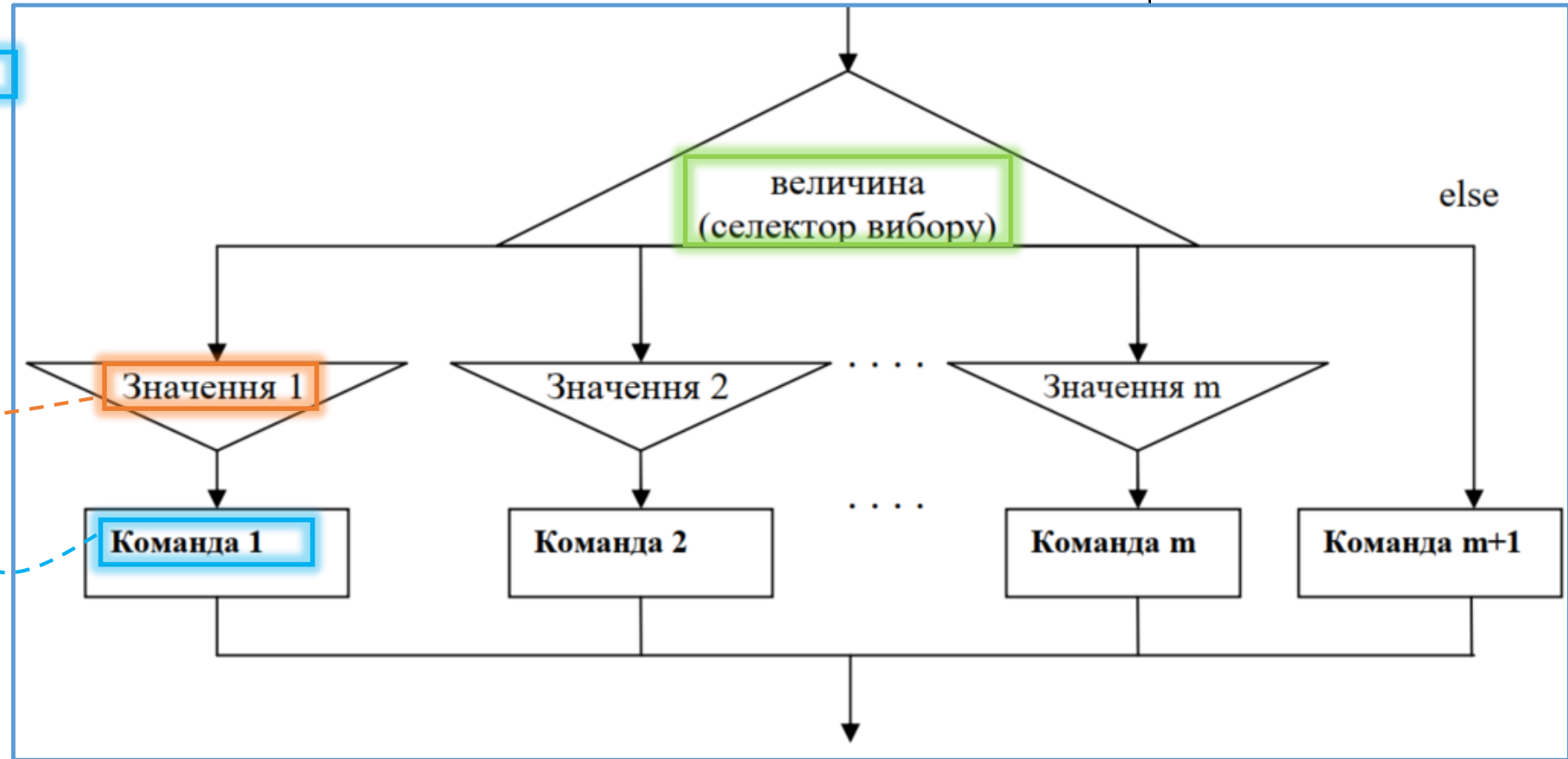
Це більш наочний спосіб порівняти вираз (в сенсі оператора «===») відразу з декількома варіантами.

```
switch (селектор) {
```

```
  case значення 1 : команда 1
```

```
    break;
```

```
}
```

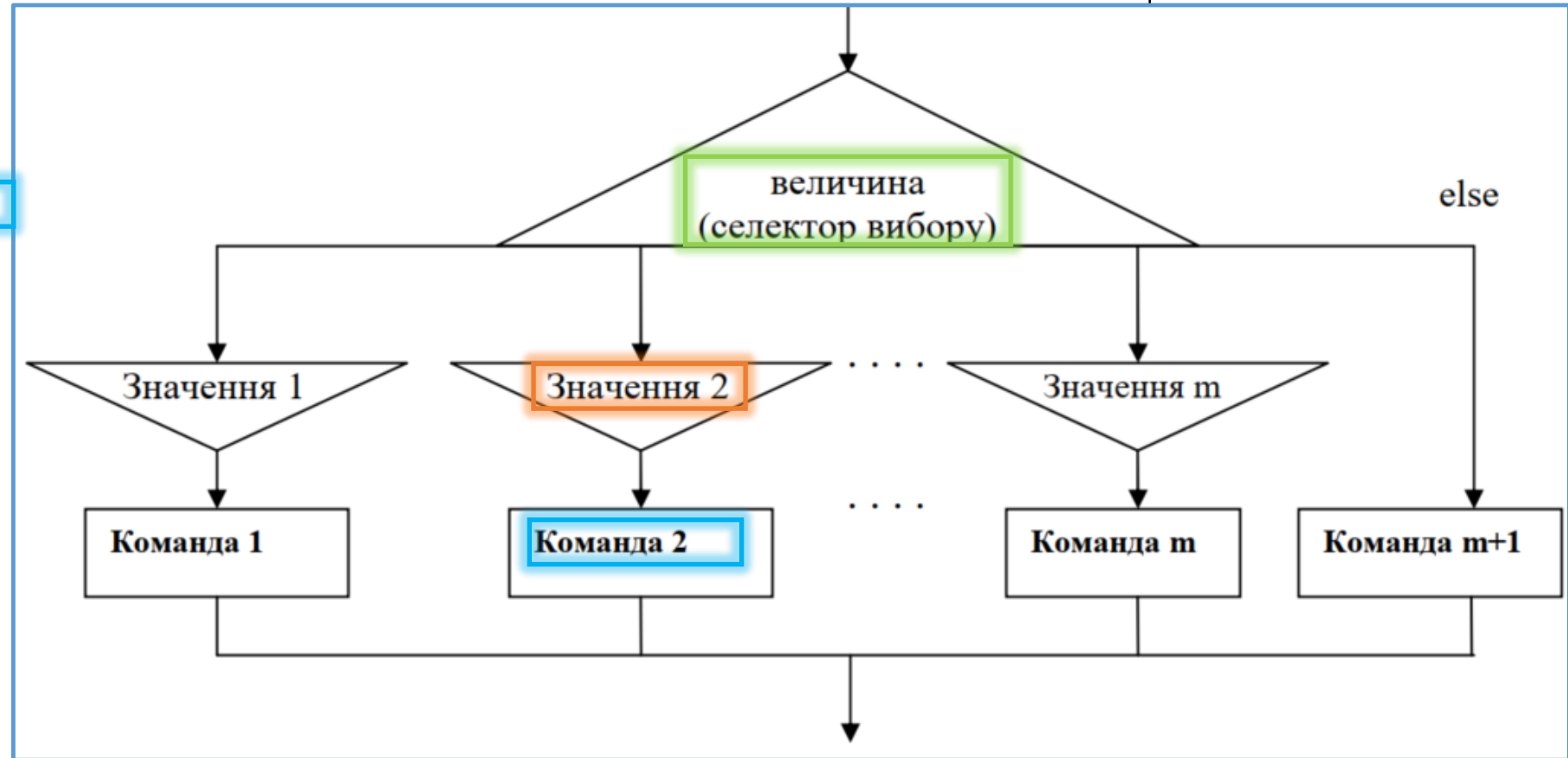


# ОПЕРАТОР ВИБОРУ SWITCH

Конструкція switch замінює собою відразу кілька if. Це оператор викорситовується тоді, коли у залежності від значення деякої величини (кількість можливих значень є невеликою) потрібно виконати ті чи інші оператори.

Це більш наочний спосіб порівняти вираз (в сенсі оператора «===») відразу з декількома варіантами.

```
switch (селектор) {  
    case значення 1 : команда 1  
        break;  
    case значення 2 : команда 2  
        break;  
    ...  
}
```

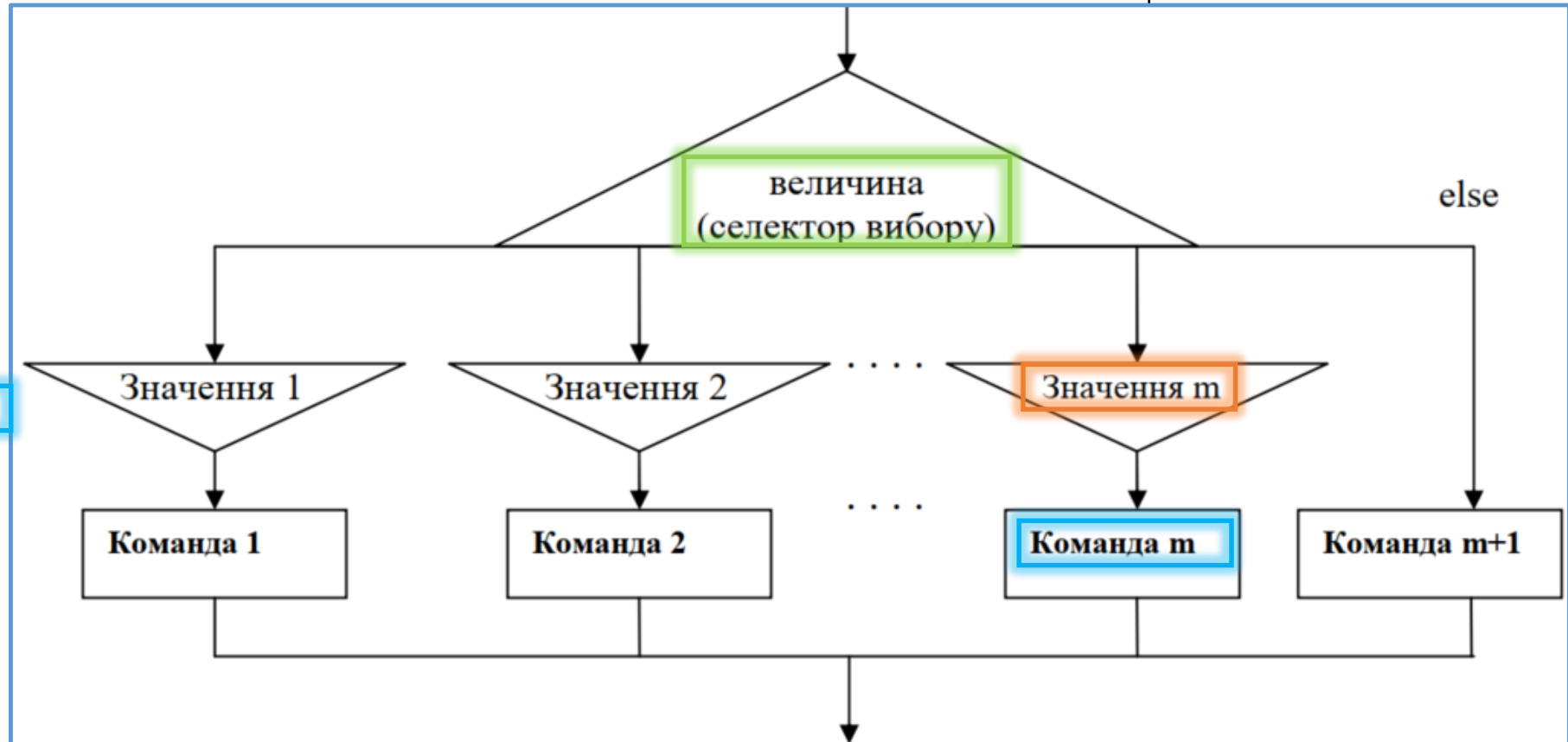


# ОПЕРАТОР ВИБОРУ SWITCH

Конструкція switch замінює собою відразу кілька if. Це оператор викорситовується тоді, коли у залежності від значення деякої величини (кількість можливих значень є невеликою) потрібно виконати ті чи інші оператори.

Це більш наочний спосіб порівняти вираз (в сенсі оператора «===») відразу з декількома варіантами.

```
switch (селектор) {  
    case значення 1 : команда 1  
        break;  
    case значення 2 : команда 2  
        break;  
    . . . . .  
    case значення m : команда m  
        break;  
}
```

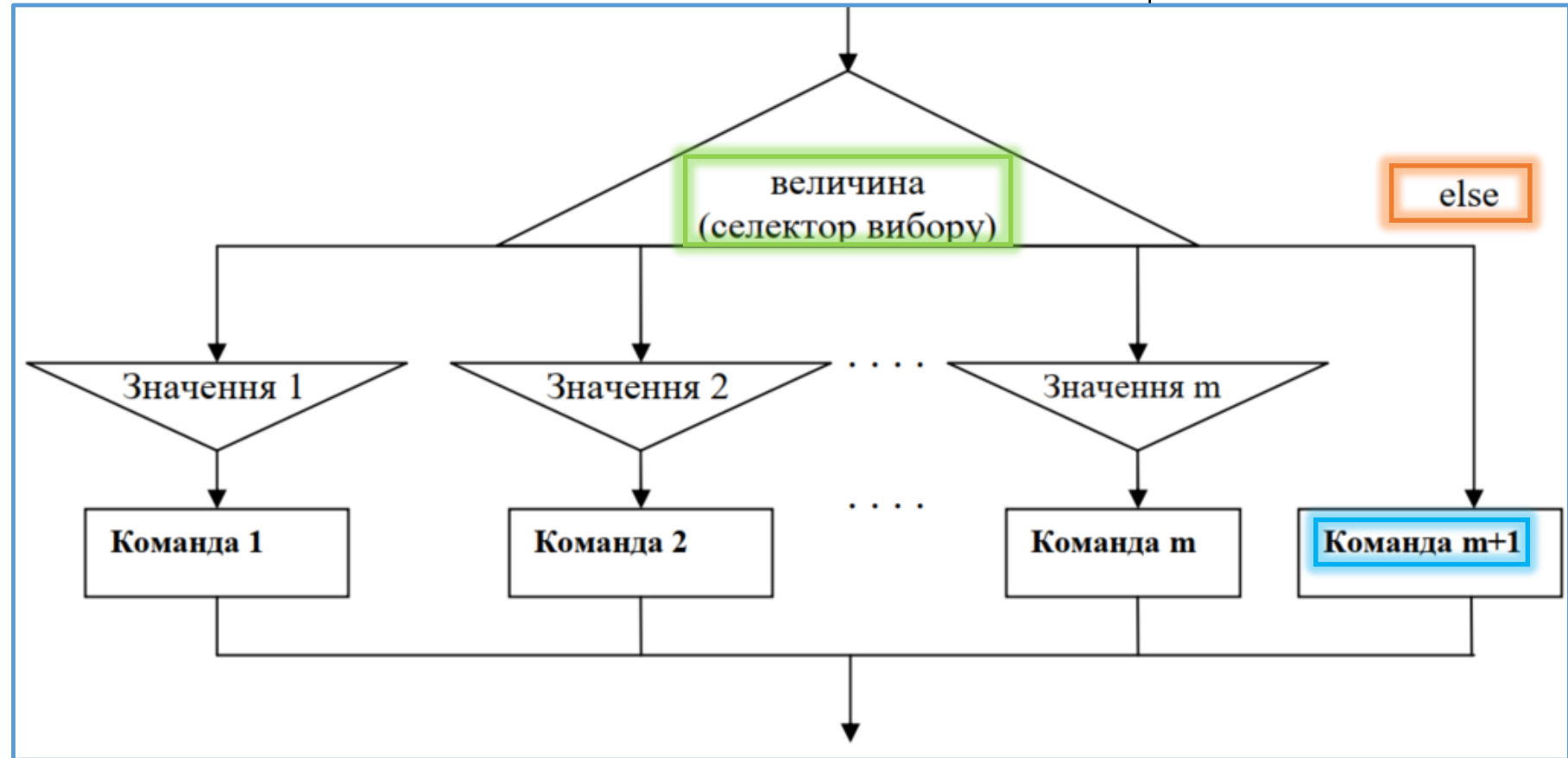


# ОПЕРАТОР ВИБОРУ SWITCH

Конструкція switch замінює собою відразу кілька if. Це оператор викорситовується тоді, коли у залежності від значення деякої величини (кількість можливих значень є невеликою) потрібно виконати ті чи інші оператори.

Це більш наочний спосіб порівняти вираз (в сенсі оператора «===») відразу з декількома варіантами.

```
switch (селектор) {  
    case значення 1 : команда 1  
        break;  
    case значення 2 : команда 2  
        break;  
    . . . . .  
    case значення m : команда m  
        break;  
    default:  
        команда m+1  
}
```



# ОПЕРАТОР ВИБОРУ SWITCH

Конструкція switch замінює собою відразу кілька if. Це оператор викорситовується тоді, коли у залежності від значення деякої величини (кількість можливих значень є невеликою) потрібно виконати ті чи інші оператори.

Це більш наочний спосіб порівняти вираз (в сенсі оператора «===») відразу з декількома варіантами.

Рівні типи і значення

Аналог з використанням *if-else*

```
switch (селектор) {
```

```
  case значення 1 : команда 1  
    break;
```

```
  if (селектор === значення1) команда 1
```

```
}
```

# ОПЕРАТОР ВИБОРУ SWITCH

Конструкція switch замінює собою відразу кілька if. Це оператор викорситовується тоді, коли у залежності від значення деякої величини (кількість можливих значень є невеликою) потрібно виконати ті чи інші оператори.

Це більш наочний спосіб порівняти вираз (в сенсі оператора «===») відразу з декількома варіантами.

Аналог з використанням *if-else*

```
switch (селектор) {  
    case значення 1 : команда 1  
        break;  
    case значення 2 : команда 2  
        break;  
}
```

```
if (селектор === значення1) команда 1  
  
else if (селектор === значення2) команда 2
```

# ОПЕРАТОР ВИБОРУ SWITCH

Конструкція switch замінює собою відразу кілька if. Це оператор викорситовується тоді, коли у залежності від значення деякої величини (кількість можливих значень є невеликою) потрібно виконати ті чи інші оператори.

Це більш наочний спосіб порівняти вираз (в сенсі оператора «===») відразу з декількома варіантами.

Аналог з використанням *if-else*

```
switch (селектор) {  
    case значення 1 : команда 1  
        break;  
    case значення 2 : команда 2  
        break;  
    . . . . .  
    case значення m : команда m  
        break;  
}
```

```
if (селектор === значення1) команда 1  
  
else if (селектор === значення2) команда 2  
  
else if (селектор === значення_m) команда m
```

# ОПЕРАТОР ВИБОРУ SWITCH

Конструкція switch замінює собою відразу кілька if. Це оператор викорситовується тоді, коли у залежності від значення деякої величини (кількість можливих значень є невеликою) потрібно виконати ті чи інші оператори.

Це більш наочний спосіб порівняти вираз (в сенсі оператора «===») відразу з декількома варіантами.

Аналог з використанням *if-else*

```
switch (селектор) {  
    case значення 1 : команда 1  
        break;  
    case значення 2 : команда 2  
        break;  
    . . . . .  
    case значення m : команда m  
        break;  
  
    default:  
        команда m+1  
}
```

```
if (селектор === значення1) команда 1  
  
else if (селектор === значення2) команда 2  
  
else if (селектор === значення_m) команда m  
  
else команда m+1
```



**Загальна форма**

**Приклад.** Вводиться оцінка – цифра, вивести оцінку  
прописом (селектор вибору цілого типу).

## Загальна форма

Приклад. Вводиться **оцінка** – цифра, вивести оцінку прописом (селектор вибору цілого типу).

```
var score = parseInt(prompt("score", ""));  
var result;
```

## Загальна форма

```
switch (<селектор вибору>
{

}
}
```

Приклад. Вводиться оцінка – цифра, вивести оцінку прописом (селектор вибору цілого типу).

```
var score = parseInt(prompt("score", ""));
var result;
//----- Знаходимо результат за допомогою switch
switch (score)
{

}
}
```

## Загальна форма

```
switch (<селектор вибору>)  
{  
    case <знач. 1> : <оператор 1> ;  
                    break;  
  
}
```

Приклад. Вводиться оцінка – цифра, вивести оцінку прописом (селектор вибору цілого типу).

```
var score = parseInt(prompt("score", ""));  
var result;  
//----- Знаходимо результат за допомогою  
switch  
    switch (score)  
    {  
        case 2: result="Незадовільно";  
                break;  
  
    }
```

## Загальна форма

```
switch (<селектор вибору>)  
{  
    case <знач. 1> : <оператор 1>;  
                    break;  
    case <знач. 2> : <оператор 2>;  
                    break;  
  
}
```

Приклад. Вводиться оцінка – цифра, вивести оцінку прописом (селектор вибору цілого типу).

```
var score = parseInt(prompt("score", ""));  
var result;  
//----- Знаходимо результат за допомогою  
switch  
    switch (score)  
    {  
        case 2: result="Незадовільно";  
                break;  
        case 3: result="Задовільно.";   
                break;  
    }  
}
```

## Загальна форма

```
switch (<селектор вибору>)  
{  
    case <знач. 1> : <оператор 1>;  
                    break;  
    case <знач. 2> : <оператор 2>;  
                    break;  
    .....  
  
}
```

Приклад. Вводиться оцінка – цифра, вивести оцінку прописом (селектор вибору цілого типу).

```
var score = parseInt(prompt("score", ""));  
var result;  
//----- Знаходимо результат за допомогою  
switch  
    switch (score)  
    {  
        case 2: result="Незадовільно";  
                break;  
        case 3: result="Задовільно.";   
                break;  
        case 4: result="Добре";  
                break;  
    }  
}
```

## Загальна форма

```
switch (<селектор вибору>)  
{  
    case <знач. 1> : <оператор 1>;  
                    break;  
    case < знач. 2> : <оператор 2>;  
                    break;  
    .....  
    case < знач. N> : <оператор N>;  
                    break;  
  
}
```

Приклад. Вводиться оцінка – цифра, вивести оцінку прописом (селектор вибору цілого типу).

```
var score = parseInt(prompt("score", ""));  
var result;  
//----- Знаходимо результат за допомогою  
switch  
    switch (score)  
    {  
        case 2: result="Незадовільно";  
                break;  
        case 3: result="Задовільно.";   
                break;  
        case 4: result="Добре";  
                break;  
        case 5: result="Відмінно";  
                break;  
    }  
}
```

## Загальна форма

```
switch (<селектор вибору>)  
{  
    case <знач. 1> : <оператор 1>;  
                    break;  
    case <знач. 2> : <оператор 2>;  
                    break;  
    .....  
    case <знач. N> : <оператор N>;  
                    break;  
    default : <оператор N+1>;  
              break;  
}
```

Приклад. Вводиться оцінка – цифра, вивести оцінку прописом (селектор вибору цілого типу).

```
var score = parseInt(prompt("score", ""));  
var result;  
//----- Знаходимо результат за допомогою  
switch  
    switch (score)  
    {  
        case 2: result="Незадовільно";  
                break;  
        case 3: result="Задовільно.";   
                break;  
        case 4: result="Добре";  
                break;  
        case 5: result="Відмінно";  
                break;  
        default: result="Неправильна оцінка.";  
                break;  
    }
```



<u><b>Загальна форма</b></u>	<u><b>Приклад.</b></u> Вводиться оцінка – цифра, вивести оцінку прописом (селектор вибору цілого типу).
<pre>switch (&lt;селектор вибору&gt; {     case &lt;знач. 1&gt; : &lt;оператор 1&gt;;                     break;     case &lt; знач. 2&gt; : &lt;оператор 2&gt;;                     break;     .....     case &lt; знач. N&gt; : &lt;оператор N&gt;;                     break;     default : &lt;оператор N+1&gt;;               break; }</pre>	<pre>var score = parseInt(prompt("score", "")); var result; //----- Знаходимо результат за допомогою switch switch {     switch (score)     {         case 2: result="Незадовільно";                 break;         case 3: result="Задовільно.";                 break;         case 4: result="Добре";                 break;         case 5: result="Відмінно";                 break;         default: result="Неправильна оцінка.";                 break;     }     alert(result); }</pre>

Задача. З клавіатури вводиться номер місця, яку зайняв спортсмен. Вивести, яку медаль він отримає:

1-золота

2-срібна

3-бронзова

Все інше - грамота

З клавіатури вводиться колір помідора: “red”, “yellow”, “green”. Вивести на екран у якому стані цей помідор:

- “red” - можна їсти,
- “yellow” - дозріває
- “green” - ще росте

Якщо для декількох варіантів необхідно виконати одні і ті ж оператори, то ці оператори вказують тільки для одного з варіантів, а для всіх інших не вказуємо ні необхідних операторів, ні операторів `break`.

**Приклад.** З клавіатури вводиться оцінка у національній шкалі, необхідно вивести повідомлення про те, чи зараховано студенту залік.

```
<script>
    // З клавіатури вводиться номер місяця, вивести на екран пору року

    var score = parseInt(prompt("score", ""));
    var result;
    //----- Знаходимо результат за допомогою switch
    switch (score)
    {
        case 1:
        case 2: result="Незараховано";
                break;
        case 3:
        case 4:
        case 5: result="Зараховано";
                break;
        default: result="Неправильна оцінка.";
                break;
    }

    //-----
    alert(result);
</script>
```

Приклад. З клавіатури вводиться номер дня (1-7). Вивести яким є цей день: робочий чи вихідний

Слід зазначити, що на відміну від інших С-подібних мов програмування в якості можливих значень селектора вибору можуть бути величини не тільки порядкового, а і інших типів. Більше того, в якості можливих значень можуть бути використано результат виконання деякого оператора.

**Приклад .**

```
var b= 1;
var a = 2;
switch (a) {
    case 1: document.write(a);
        break;
    case 3.5: document.write(a);    ← Значення є дійсним числом 3.5
        break;
    case b+1: document.write(a);    ← Значення обраховується «b+1»
        break;
}
```

## КОРОТКИЙ ЦИКЛ ОБЧИСЛЕНЬ

`змінна` = `вираз_1` || `вираз_2` || `вираз_3` || ... || `вираз_N`

- результатом обчислення буде значення першого виразу, який еквівалентний `true` (усі інші вирази праворуч взагалі не обчислюються),
- якщо усі значення еквівалентні `false`, то результат буде дорівнювати значенню

останнього виразу `вираз_N`

Приклад.

`b=0, c=undefined, d=45`

`a = b || c || d` // `a=45`

`a = b || d || c` // `a=45`

`a = d || b || c` // `a=45`

`a = b || c` // `a=undefined`

---

## КОРОТКИЙ ЦИКЛ ОБЧИСЛЕНЬ

`змінна` = `вираз_1` ?? `вираз_2` ?? `вираз_3` ?? ... ?? `вираз_N`

- результатом обчислення буде значення першого виразу, який не є `null` або `undefined` (усі інші вирази праворуч взагалі не обчислюються),
- якщо усі значення `null` або `undefined`, то результат буде дорівнювати значенню останнього виразу `вираз_N`

Приклад.

`b=0, c=undefined, d=45, f=null`

`a = b ?? c // a=0`

`a = c ?? d // a=45`

`a = c ?? f ?? b // a=0`

**Операція умовного присвоєння** «`??=`». Присвоєння, якщо поточне значення змінної `null` або `undefined`

`a=null, b=undefined, c=7`

`a??=9 // a=9`

`b??=3 //b=3`

`c??=5 //c=7`



## КОРОТКИЙ ЦИКЛ ОБЧИСЛЕНЬ

`змінна` = `вираз_1` && `вираз_2` && `вираз_3` && ... && `вираз_N`

- результатом обчислення буде значення першого виразу, який еквівалентний `false`, (усі інші вирази праворуч взагалі не обчислюються),
- якщо усі значення еквівалентні `true`, то результат буде дорівнювати значенню

останнього виразу `вираз_N`

Приклад.

`b=0, c=undefined, d=45`

`a = b && c && d // a=0`

`a = d>b && d && b==0 // a=true`

`a = d && c && b // a= undefined`