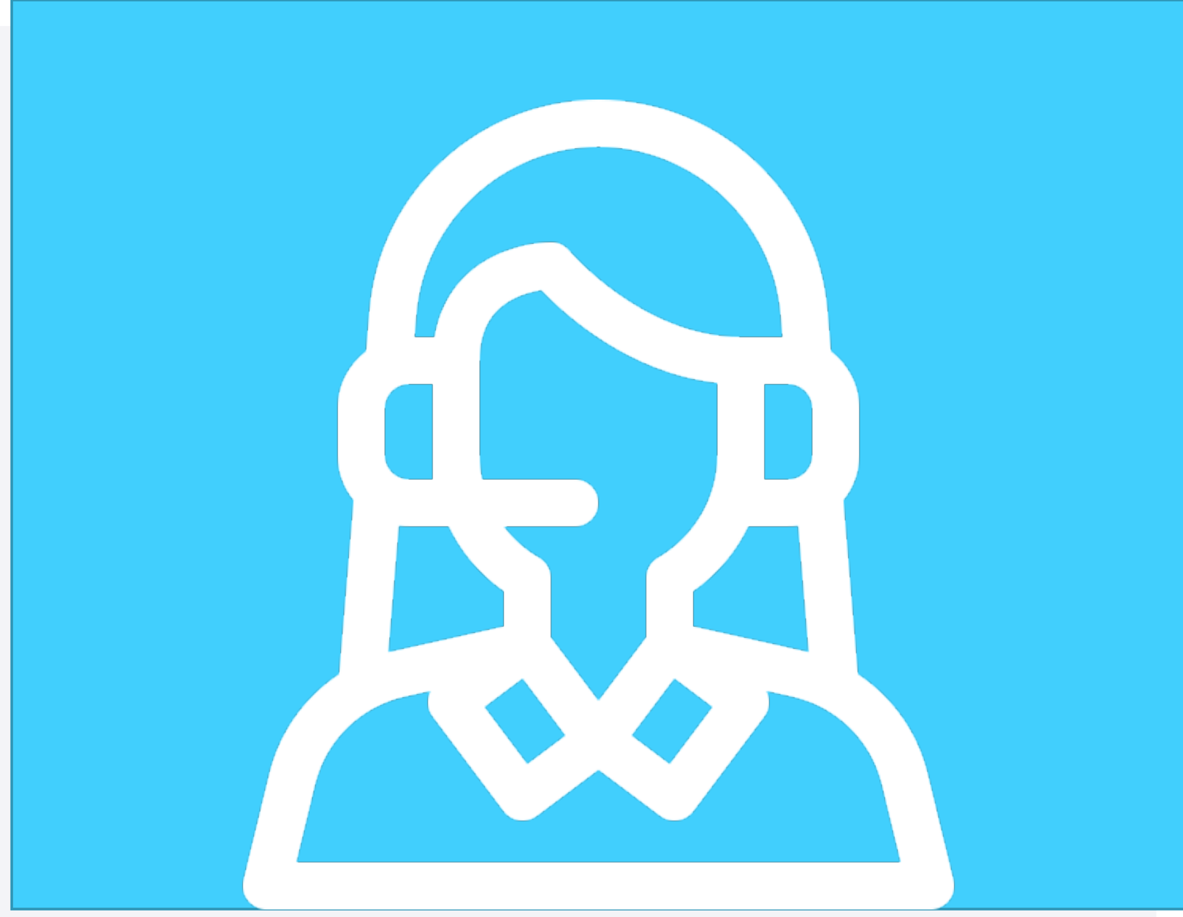


Conditional Statements

Logical expressions and checks.
Conditional statement if-else.

Content

Content

01

Comparison operators

Logical expressions and checks

Comparison operators

Operator	Sign
Equal (value, type and value)	`==`, `===`
Not equal (value, type and value)	`!=`, `!==`
Greater than	`>`
Greater than or equal	`>=`
Less than	`<`
Less than or equal	`<=`

Compare values

In programming, we can compare values
The result of logical expressions is true or false

```
let a = 10;
let b = 5;
console.log( a > b );               // true
console.log( a > 0 );               // true
console.log( a > 100 );             // false
console.log( a < a );               // false
console.log( a <= 10 );             // true
console.log( b == 50 / a );         // true
```

03

Conditional statements

Simple checks

Simple checks

We often check conditions and perform actions according to the result

Condition (Boolean expression)

Condition True Execution Code

```
if (...) {
  // Execution code
}
```

The result of the check is true or false

Grade

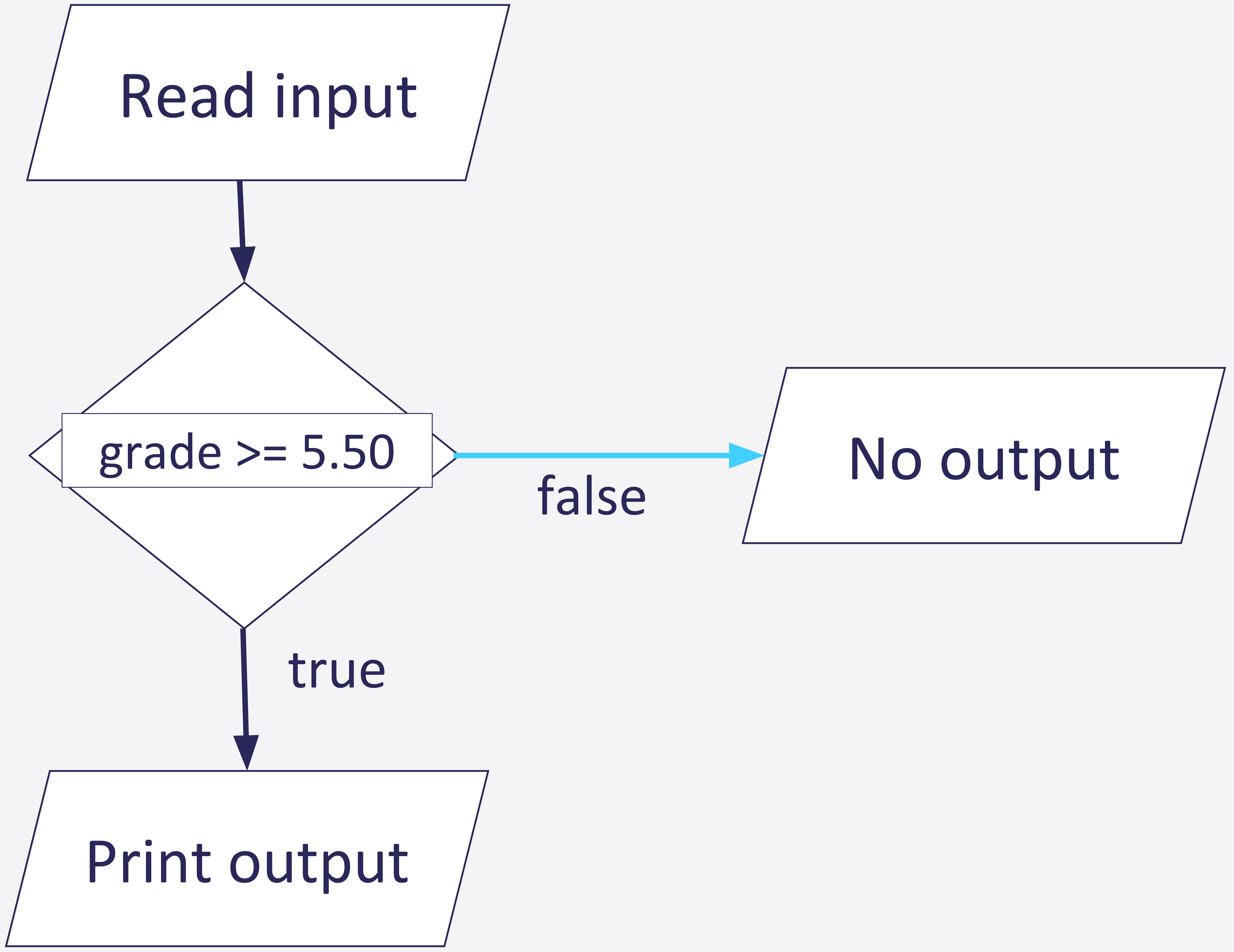
Write a function that:

- Read a grade (number)
- It checks to see if it's excellent.
- Prints "Excellent!" on the console if the rating is greater than or equal to 5.50

Example:

4	no output
5.50	Excellent!

Block diagram



if-else

In case of falseness of the condition, we can perform other actions – through the else construction

```
if (...) {
    // Execution code
} else {
    // Execution code
}
```

Condition Incorrect Execution Code

The Higher Number – Condition

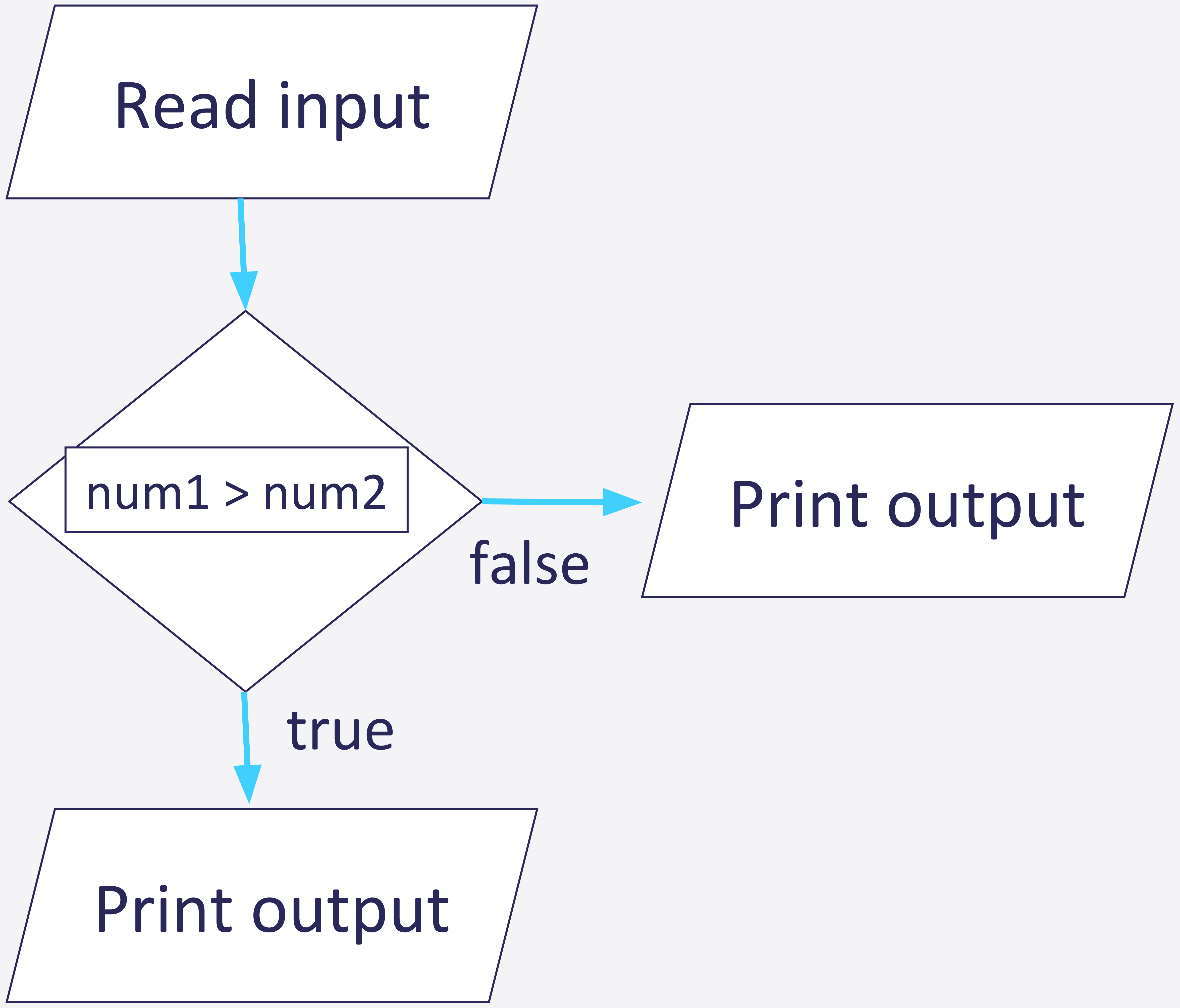
Write a program that:

- Receives two integers
- Displays "Greater number: "
- Prints the larger of them on the console

Example:

Input		Output
6 8	→	8
5 3	→	5

Block diagram



Sirma

Series of checks

The if/else-if/else… is a series of checks

```
if (...)
// Execution code
else if (...)
// Execution code
else if (...)
// Execution code
```

If a condition is true, do not proceed to check the following conditions

Series of checks - example

The program checks the first condition, establishes, that it's true and the execution ends.

```
let a = 9;
if (a > 4)
   console.log("Bigger than 4");
else if (a > 5)
   console.log("Bigger than 5");
else
   console.log("Equal to 9");
```

Displays only "Bigger than 4" on the console

Numbers 0 to 9 with text

Write a function that:

- Receives an integer
- Checks its value [0.9]
  - If the number is greater than 9 prints "too big"
- Prints the value with text

Example:

```
7   →   seven          10   →   too big
```

Numbers 0 to 9 with text

```
if (num == 0)
    console.log("zero");
else if (num == 2)
    console.log("two");
else if (num > 9)
    console.log("too big");
```

Variable Scope

Range within which it can be used
Example: The variable price exists only in the if-construct code block

```
let day = "Monday";
if (day == "Monday") {
    let price = 5;
}

console.log(price); // Error
```

Face on figure

Write a function that:
Receives the appearance of a geometric figure ("square", "rectangle", "circle" or "triangle")
Calculates the face according to the type of figure
Sample input and output:

```
square, 5  →  25
```

```
rectangle, 10, 3.5  →  35
```

sirma.com

Conditional Statement switch-case

Works as a series if/else if/else if…

```
switch (...) {
  case ...:
     // code
     break;
  case ...:
     // code
     break;
  default:
     // code
     break;
}
```

List conditions (values) for the check

The condition in switch case is variable

Code that will run if there is no match with any case

sirma.com

Multiple cases in switch-case

Through switch-case, we can execute the same code for multiple conditions

The code will be executed if any of the three conditions in the series is true

```
switch (...) {
  case ...:
  case ...:
  case ...:
    // code
    break;
default:
    // code
    break;
}
```

04

Debugging

Simple debugger operations

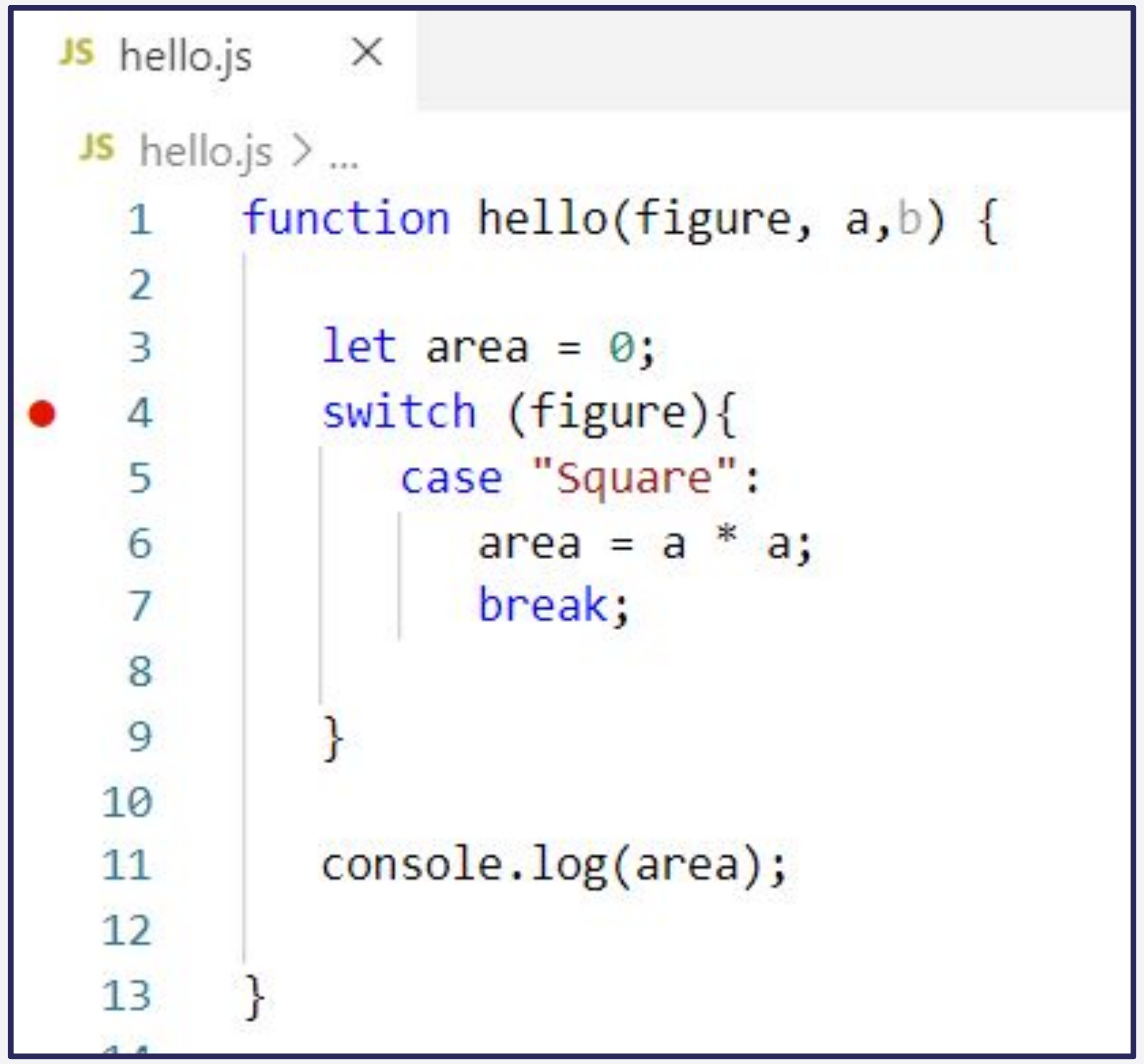
Sirma

Debugging

Process of monitoring the implementation of the program

This allows us to detect errors in the code (bugs)

Breakpoint



```
function hello(figure, a,b) {

   let area = 0;
   switch (figure){
     case "Square":
       area = a * a;
       break;

   }

   console.log(area);

}
```

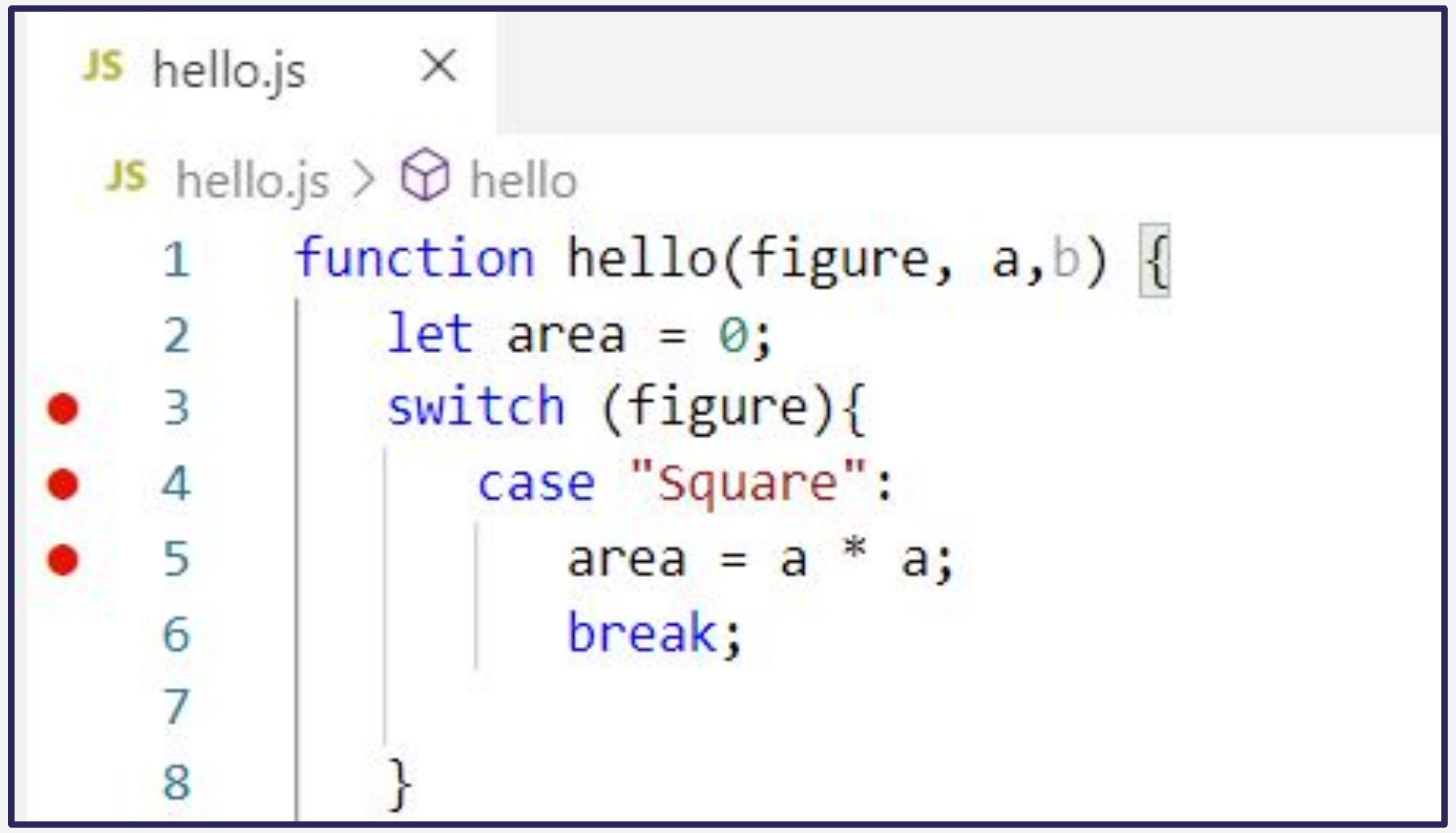
Sirma

Debugging

Pressing [F5] will start the program in debug mode

We can move on to the next step with [F10]

We can create [F9] stoppers – breakpoints
We can get to them directly using [F9]



```
function hello(figure, a,b) {
  let area = 0;
  switch (figure){
    case "Square":
      area = a * a;
      break;

  }
```

05

Nested conditional statements

Nested conditional statements

Only when the first condition is met the nested check is reached

```
if (condition1){
    console.log("condition1 valid");
    if (condition2)
        console.log("condition2 valid");
    else
        console.log("condition2 not valid");
}
```

Nested if construction

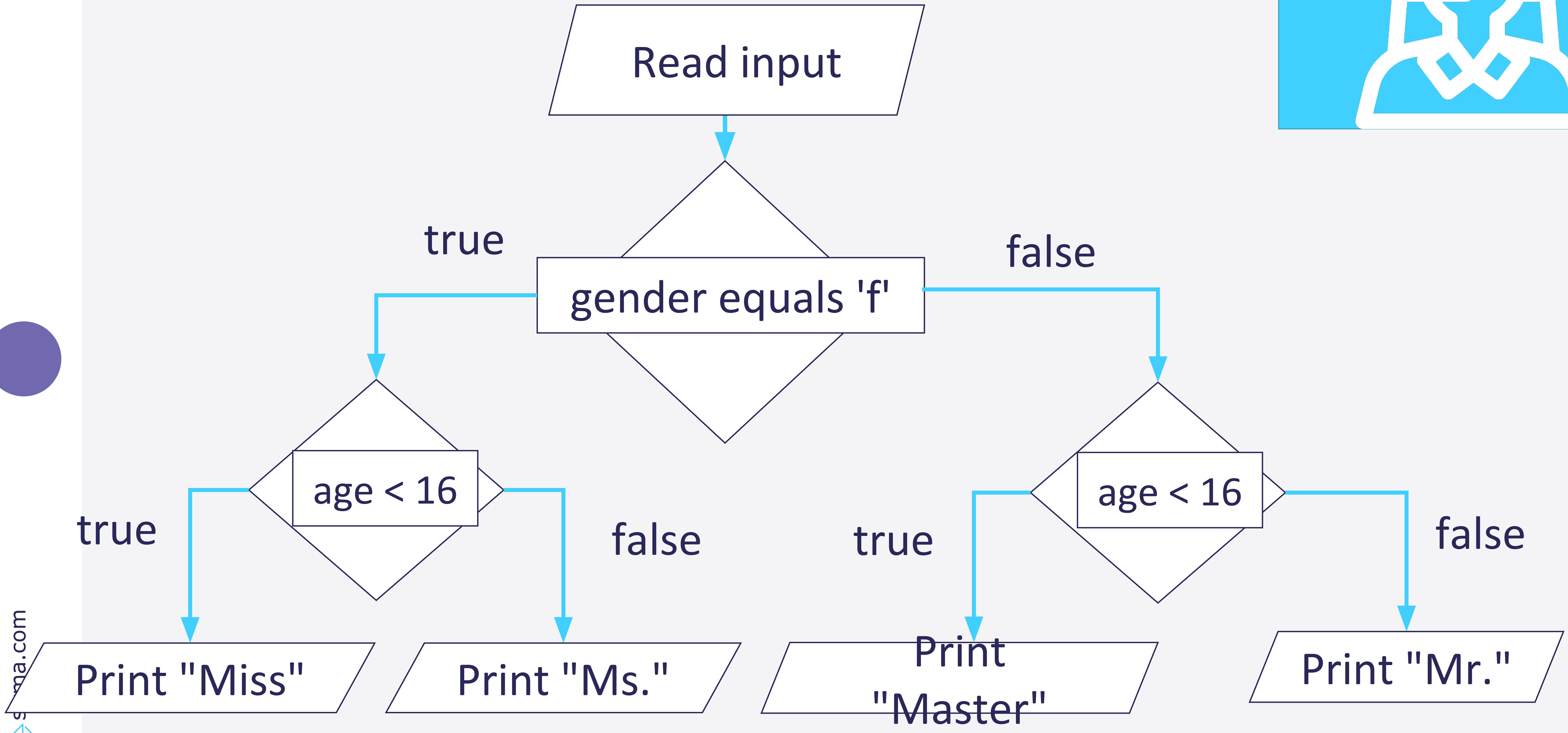
Address by age and gender

Write a function that receives:
Age
Gender
Prints an address according to the data entered, as shown in the diagram (in the next slide)
Sample input and output:

```
12        Miss
f    →

16        Mr.
m    →
```

Block diagram



Grocery store - Task

Write a function that:
Receives as parameters:
Product name, city and quantity
Calculate its price according to the table:

City/product	coffee	water	juice	sweets	chips
Sofia	0.50	0.80	1.20	1.45	1.60
Plovdiv	0.40	0.70	1.15	1.30	1.50
Varna	0.45	0.70	1.10	1.35	1.55

Grocery store

Sample input and output:

```
coffee
Varna
2
```

→

```
0.9
```

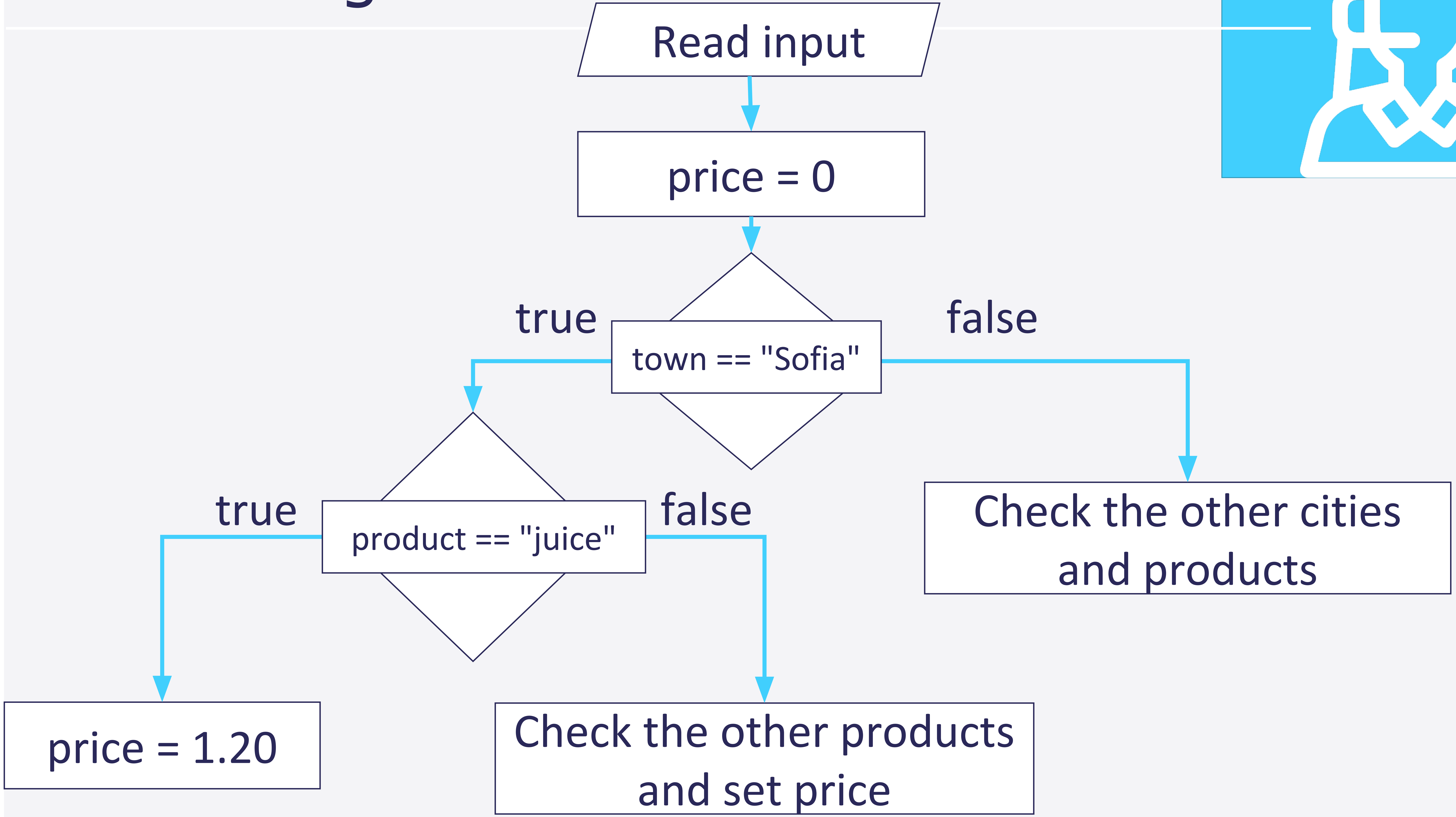
```
chips
Plovdiv
1
```

```
1.5
```

```
juice
Sofia
6
```

```
7.2
```

Block diagram



05

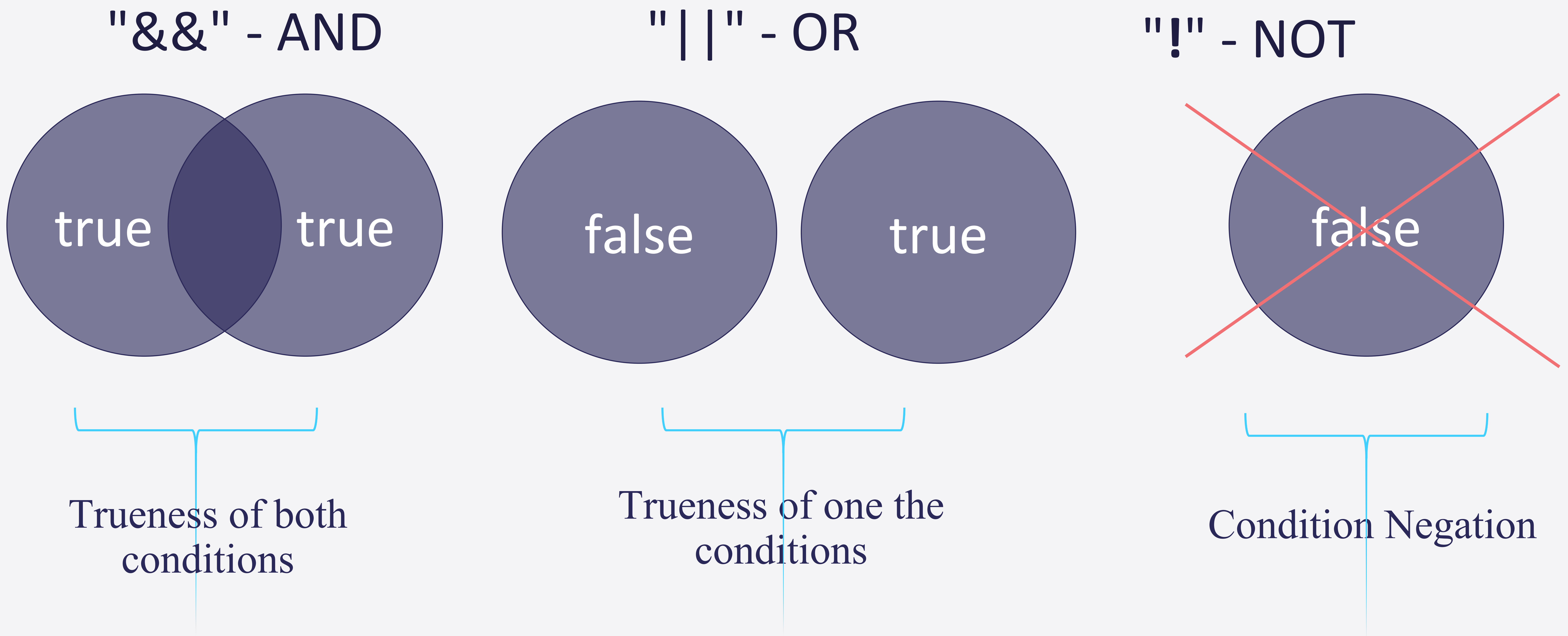
Logical operators

&&, ||, !

sirma.com

Boolean operators

Operators combining or excluding conditions
Return Boolean result (true or false)



Logical "AND"

Checks the fulfillment of several conditions at the same time
Example: check if a number is simultaneous: greater than 5 and less than 10

&&

```
let a = 5;
if ((a > 5 && a < 10) && a != 2 == 0) …
```

Logical "OR"

Checks to see if at least one of several conditions is met

Example: check if the entered word is "Test" or "Demo"

```
if (word == "Test" || word == "Demo")
```

Logical negation

Checks to see if a condition is not met
Example:
Check if a number is greater than 10 and even:

```
let number = 5;
let isValid = (number < 10) && (number > 0);
if (!isValid)
{
    Console.WriteLine("Invalid");
}
```

Priority of conditions

By parentheses () we can prioritize conditions

```
let a = 50;
let b = 200;
let c = 300;
if ((a >= 100 && b <= 200) || (c + b >= 300 && c <= 400)) {
    console.log("Yes"); // Yes
}

if (a >= 100 && (b <= 200 || c + b >= 300) && c <= 400) {
    console.log("Yes"); // No output
}
```

Vegetable market - Task

Write a function that receives product, day, quantity
Displays the price according to the day and product

On working days sell at the following prices:

vegetable	tomato	onion	lettuce	cucumber	pepper
price	2.50	1.20	0.85	1.45	5.50

On weekends, prices are higher:

vegetable	tomato	onion	lettuce	cucumber	pepper
price	2.80	1.30	0.85	1.75	3.50

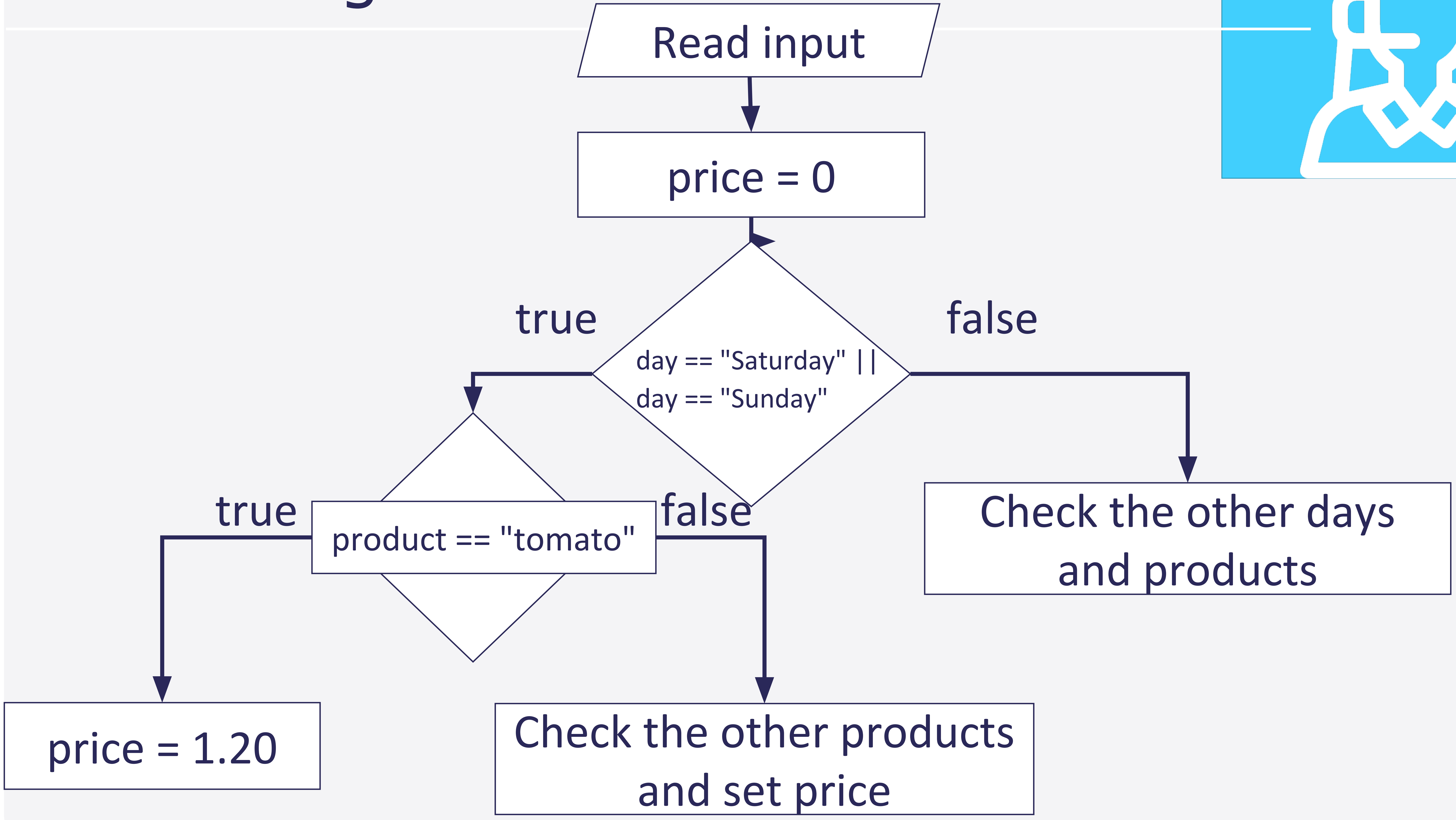
```
tomato
Tuesday
2
```

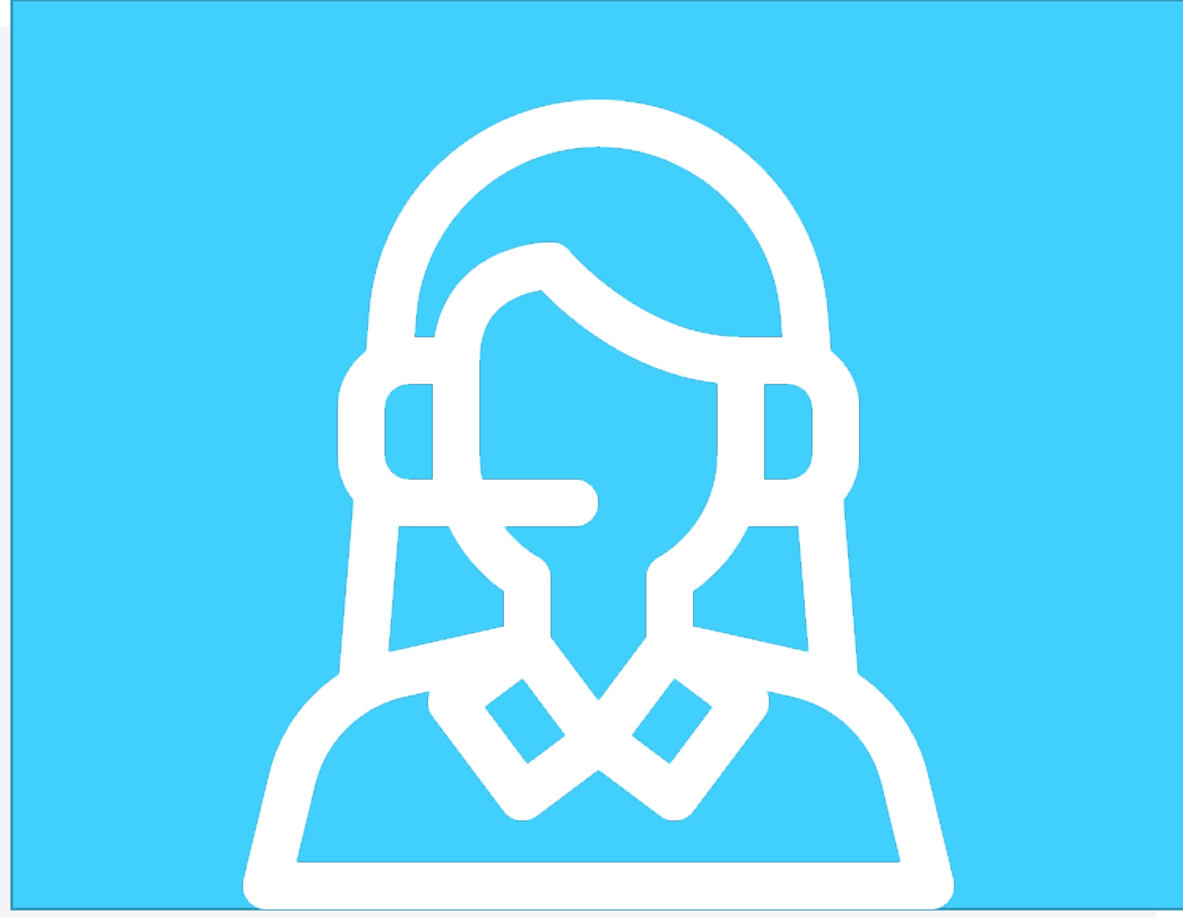
```
5.00
```

```
onion
Sunday
3
```

```
3.90
```

Block diagram





Summary

Conditional Statements
IF and IF-ELSE
Variable Scope
Debugging
Switch-case statement

Nested conditional statements:
More complex checks with &&, ||, ! and ()

Thank you!