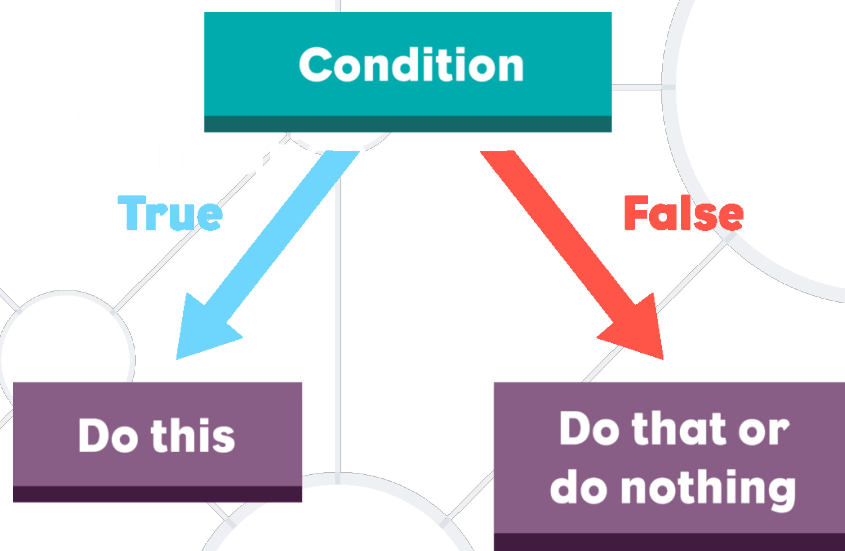


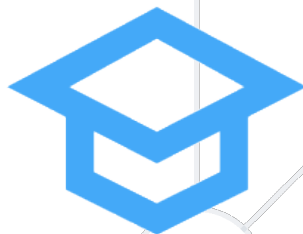
# Проверки

Логически изрази и проверки. Условна конструкция If-else



СофтУни

Преподавателски екип



SoftUni



Software University

<https://softuni.bg>

1. Преговор
2. Логически изрази и проверки
  - Оператори за сравнение
3. Условни конструкции
4. Закръгляне и форматиране
5. Серия от проверки
6. Живот на променлива
7. Решаване на изпитна задача





**Преговор**

1. Какво ще се отпечата на конзолата, ако изпълним следната команда:
- ```
console.log("a" + "b");
```

ab

Error

ba

195

2. Какъв е типът на променливата:

```
let number = "1000";
```

string

int

char

double

3. Как се нарича долепването на два текста (низа)?



Събиране

Конкатенация

Кулминация

Съединяване

4. Какво ще се отпечата на конзолата, ако изпълним следната команда:

```
console.log(10 % 3);
```

10

1

0

3

5. Каква стойност държи променливата **result**:

```
let a = 5;  
let b = 2;  
let result = a / b;
```

2

7

2.5

1



6. Какъв би бил резултатът, ако се опитаме да изпълним следната команда:

```
console.log(1 + 1 + "4" + 2 + 1);
```

Error

243

9


2421



# Логически изрази и проверки

Оператори за сравнение

# Оператори за сравнение



| Оператор                            | Означение |
|-------------------------------------|-----------|
| Равенство по стойност (и тип данни) | ==, ===   |
| Различно по стойност (и тип данни)  | !=, !==   |
| По-голямо                           | >         |
| По-голямо или равно                 | >=        |
| По-малко                            | <         |
| По-малко или равно                  | <=        |

- В програмирането можем да сравняваме стойности
  - Резултатът от логическите изрази е **true** или **false**

```
let a = 5;  
let b = 10;  
console.log(a < b);           // true  
console.log(a > 0);           // true  
console.log(a > 100);         // false  
console.log(a < a);           // false  
console.log(a <= 5);          // true  
console.log(b == 2 * a);      // true  
console.log("2" === 2);       // false
```



- **boolean** – булева променлива се инициализира като всички останали
- Може да има само следните две стойности: **true** (вярно) или **false** (грешно)

```
let isValid = true;
```

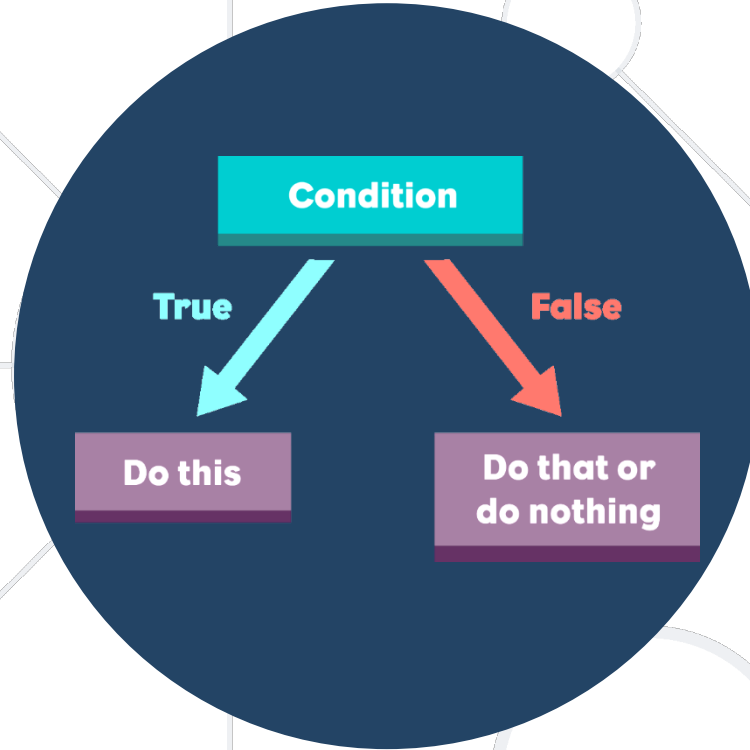
- Може да се създаде и с условие, което се свежда до true или false

```
let isPositive = a > 0;
```

# Булева променлива - Пример

```
let a = 5;  
let isPositive = a > 0;  
console.log(isPositive); // true
```

```
let a = -5;  
let isPositive = a > 0;  
console.log(isPositive); // false
```



# Условни конструкции

Прости проверки

# Прости проверки

- Често проверяваме условия и извършваме действия според резултата



Условие (булев израз)

```
if (...) {  
    // код за изпълнение  
}
```

Код за изпълнение  
при вярност на  
условието

- Резултатът е **true** или **false**



- Напишете **функция**, която:
  - **Получава** оценка (**число**)
  - **Проверява** дали е отлична
  - **Отпечатва на конзолата** "Excellent!", ако оценката е **по-голяма или равна на 5.50**
- Пример:

4 → няма изход

5.50 → Excellent!

**Read input**

**grade  $\geq$  5.50**

**No output**

false

true

**Print output**



# Прости проверки – if-else

- При **невярност** (**false**) на условието, можем да изпълним други действия – чрез **else** конструкция



```
if (...) {  
    // код за изпълнение  
} else {  
    // код за изпълнение  
}
```

Код за изпълнение  
при невярност на  
условието

- Къдравите скоби { } въвеждат блок от код (група команди)
- Ако конструкцията **if** няма скоби, се изпълнява само следващият ред

```
let color = "red";  
if (color === "red")  
    console.log("tomato");  
else  
    console.log("banana");  
console.log("bye");
```

Изпълнява се винаги – не е част от if/else конструкцията

```
let color = "red";  
if (color === "red") {  
    console.log("tomato");  
} else {  
    console.log("banana");  
    console.log("bye");  
}
```

# По-голямото число – условие

- Напишете функция, която:
  - Получава две **числа**
  - Извежда "Greater number: "
  - Отпечатва на конзолата **по-голямото** от тях
- Пример:

5  
8



8

7  
3



7



Read input

$\text{num1} > \text{num2}$

Print output

false

true

Print output

# Четно или нечетно число – условие

- Напишете функция, която:
  - Проверява, дали едно число е **четно** или **нечетно**
  - Ако е четно, отпечатва на конзолата **"even"**
  - Ако е нечетно, отпечатва на конзолата **"odd"**
- Пример:

4 → even

7 → odd

# Четно или нечетно – решение

```
function isEven(input) {  
  num = Number(input[0])  
  if (num % 2 == 0) {  
    console.log("even");  
  } else {  
    console.log("odd");  
  }  
}
```

Тестване на решението: <https://judge.softuni.bg/Contests/Index/2401#2>





# Дебъгване

Прости операции с дебъгер

# Дебъгване

- Процес на проследяване на изпълнението на програмата
- Това ни позволява да откриваме грешки (бъгове)



Breakpoint

```
1  function solve() {  
2      let currentDay = "Monday";  
3      let salary = 0;  
4      if (currentDay === "Monday") {  
5          salary = 100;  
6      }  
7      console.log(salary);  
8  }
```

# Дебъгване във Visual Studio Code

- Натискане на **[F5]** ще стартира програмата в debug режим
- Можем да преминем към следващата стъпка с **[F10]**
- Можем да създаваме **[F9]** стопери – breakpoints
  - До тях можем директно да стигнем използвайки **[Shift + F11]**





**Закръгляне на числа**

# Закръгляне на числа (1)

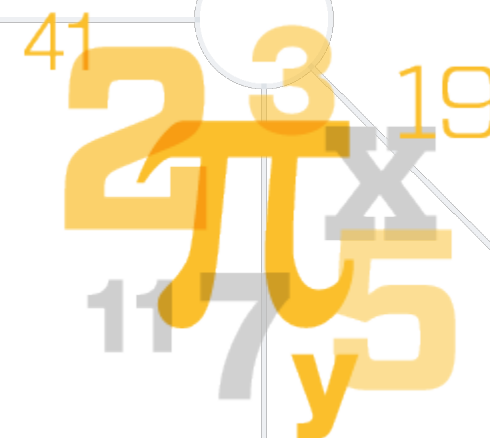
- В програмирането можем да закръгляме дробни числа

- Закръгляне до следващо (по-голямо) цяло число:

```
let up = Math.ceil(23.45); // up = 24
```

- Закръгляне до предишно (по-малко) цяло число:

```
let down = Math.floor(45.67); // down = 45
```



- Отрязване на знаците след десетичната запетая:

```
let trunc = Math.trunc(45.67); // trunc = 45
```

- Форматиране до 2 знака след десетичната запетая:

```
(123.456).toFixed(2); // 123.46
```

Брой символи след  
десетичната запетая




**Серии от проверки**

**По-сложни условни конструкции**

# Сери от проверки

- Конструкцията `if/else` - `if/else...` е серия от проверки



```
if (...)  
    // код за изпълнение  
else if (...)  
    // код за изпълнение  
else if (...)  
    // код
```


TRUE OR FALSE?

- При истинност на едно условие, **не се продължава** към проверяване на следващите условия



# Серия от проверки – пример

- Функцията проверява първото условие, установява, че е вярно и приключва



```
let a = 7;  
if (a > 4)  
    console.log("Bigger than 4");  
else if (a > 5)  
    console.log("Bigger than 5");  
else  
    console.log ("Equal to 7");
```

Извежда на  
конзолата само  
"Bigger than 4"

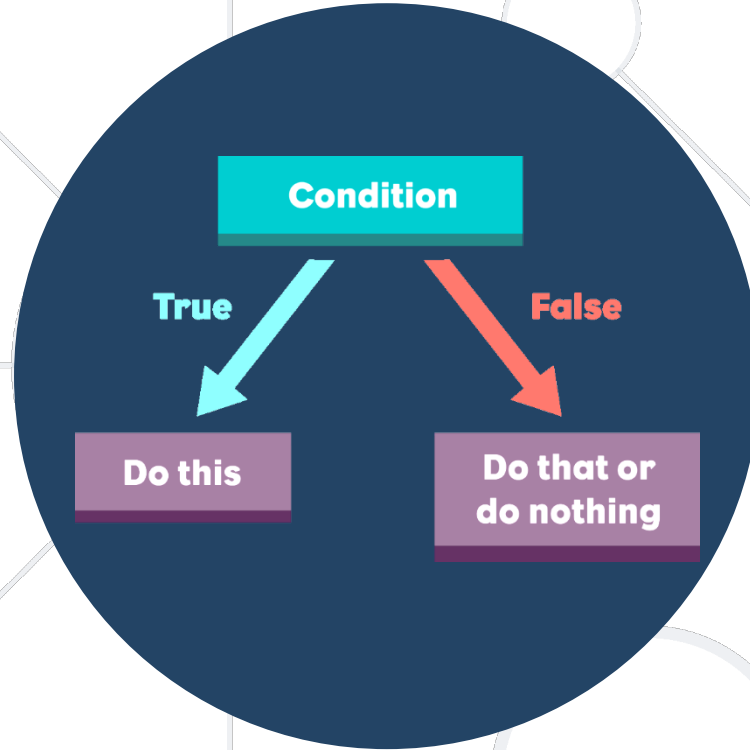


**Живот на променлива**

Диапазон на използване

- Обхват, в който може да бъде използвана
  - Пример: Променливата **salary** съществува **само** в блока от код на **if**-конструкцията

```
let currentDay = "Monday";  
if (currentDay === "Monday") {  
    let salary = Number(input);  
}  
console.log(salary); // Error!
```



# Условни конструкции

Решаване на задачи в клас (лаб)

- Напишете функция, която:
  - Получава **вид** на **геометрична фигура** ("square", "rectangle", "circle" или "triangle")
  - Пресмята **лицето** спрямо вида на фигурата
- Примерен вход и изход:

square  
5 → 25

rectangle  
7  
2.5 → 17.5

# Лица на фигури – решение

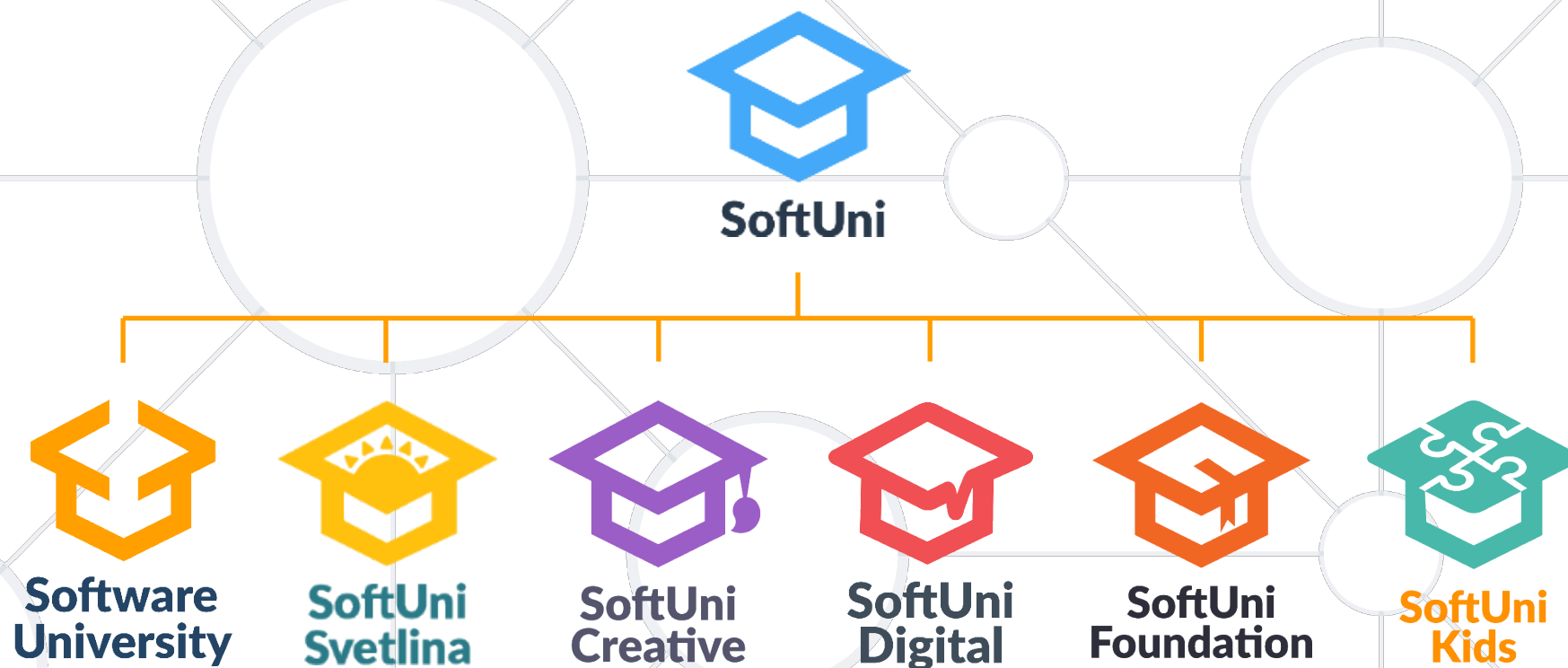
```
function areaCalculation(input){  
  let shape = input[0];  
  let area = 0;  
  if(shape === "square") {  
    let side = Number(input[1]);  
    area = side * side;  
  } else if(shape === "rectangle") {  
    let sideA = Number(input[1]);  
    let sideB = Number(input[2]);  
    area = sideA * sideB;  
  } //TODO: add more conditions  
  console.log(area);  
}
```

Тестване на решението: <https://judge.softuni.bg/Contests/Index/2401#5>

- Конструкции за проверка на условие – **if** и **if-else**
- Дебъгване
- Закръгляне и форматиране
- Живот на променливата



# Въпроси?

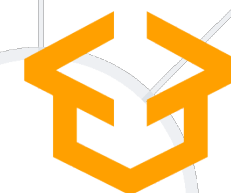




- Този курс (презентации, примери, демонстрационен код, упражнения, домашни, видео и други активи) представлява **защитено авторско съдържание**
- Нерегламентирано копиране, разпространение или използване е незаконно
- © СофтУни – <https://softuni.org>
- © Софтуерен университет – <https://softuni.bg>



- Софтуерен университет – качествено образование, професия и работа за софтуерни инженери
  - [softuni.bg](http://softuni.bg)
- Фондация "Софтуерен университет"
  - [softuni.foundation](http://softuni.foundation)
- Софтуерен университет @ Facebook
  - [facebook.com/SoftwareUniversity](https://facebook.com/SoftwareUniversity)
- Дискуссионни форуми на СофтУни
  - [forum.softuni.bg](http://forum.softuni.bg)



Software University

