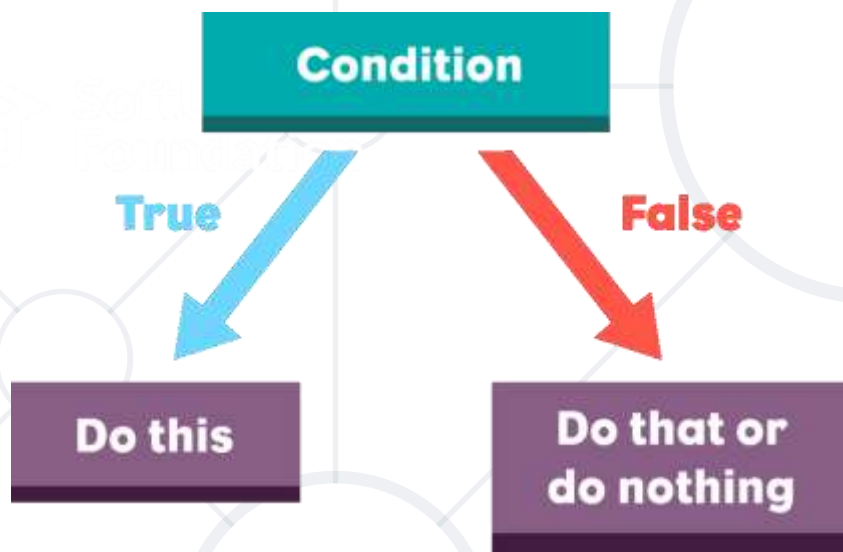


Условни конструкции

Логически изрази и проверки. Условна конструкция if-else



СофтУни

Преподавателски екип



SoftUni



Software University

<http://softuni.bg>

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





Преговор

1. Какъв е типът на променливата:

```
... number = "1000";
```

char

int

string

double

2. Какъв е типът на променливата:

```
... number = 1000;
```

int

string

char

double

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

```
cout << 10 % 3 << endl;
```

10

1

0

3

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

```
int a = 5;  
int b = 2;  
double result = a / b;
```

2.5

7

2.0

1



Логически изрази и проверки

Оператори за сравнение

Оператори за сравнение



| Оператор | Означение | Работи за |
|---------------------|-----------|---------------------------------|
| Равенство | == | числа, други сравними типове |
| Различно | != | |
| 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**

```
int a = 5;  
int b = 10;  
cout << (a < b) << endl; // 1  
cout << (a > 0) << endl; // 1  
cout << (a > 100) << endl; // 0  
cout << (a < a) << endl; // 0  
cout << (a <= 5) << endl; // 1  
cout << (b == 2 * a) << endl; // 1
```

1 = true

0 = false



- Сравняване на текст чрез оператор за равенство (==)

```
string a = "Example";  
string b = a;  
cout << (a == b) << endl;  
// 1
```

```
string a, b;  
cin >> a >> b;  
cout << (a == b) << endl; // 1
```

Въвеждане на
еднаква стойност

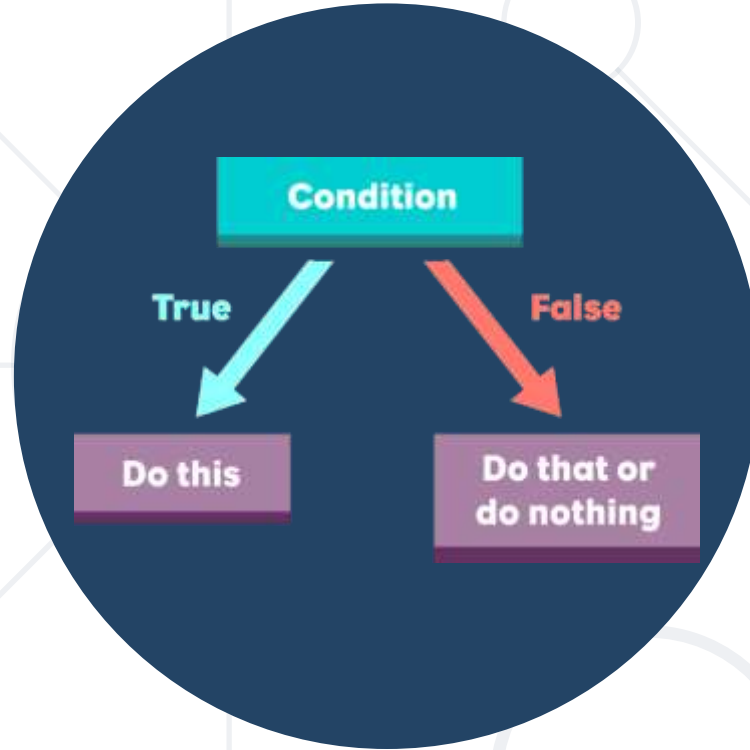
- **bool** – ключова дума, с която се инициализира булева променлива
- Има само следните две стойности **true** (вярно) или **false** (грешно)

```
bool isValid = true;
```
- Може да се създаде и с условие, което се свежда до **true** или **false**

```
bool isPositive = a > 0;
```

```
int a = 5;  
bool isPositive = a > 0;  
cout << isPositive << endl; // 1
```

```
int a = -5;  
bool isPositive = a > 0;  
cout << isPositive << endl; // 0
```



Условни конструкции

Прости проверки

Прости проверки

- Често проверяваме условия и извършваме действия според резултата **true** или **false**



Условие
(булев израз)

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

Код за изпълнение при
вярност на условието

// код за изпълнение

- Напишете **програма**, която:
 - **Чете** оценка (**число**), въведена от потребителя
 - **Проверява** дали е отлична
 - **Отпечатва на конзолата** "Excellent", ако оценката е по-голяма или равна на 5
- Пример:



Read input

grade \geq 5

No output

false

true

Print output

Прости проверки - if-else

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

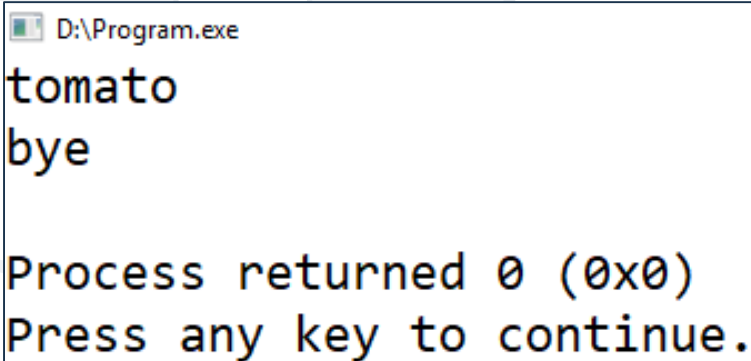


```
if (...)
{
    // код за изпълнение
}
else
{
    // код за изпълнение
}
```

Код за изпълнение
при невярност на
условието

- Къдравите скоби { } въвеждат блок (група команди)

```
string color = "red";  
if (color == "red")  
    cout << "tomato" << endl;  
else if (color == "yellow")  
    cout << "banana" << endl;  
cout << "bye" << endl;
```



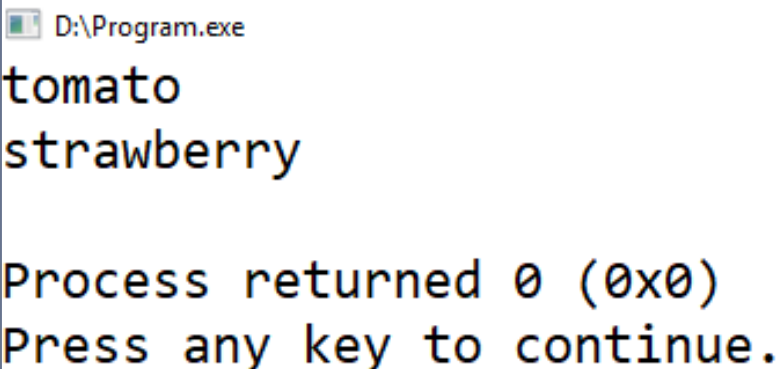
```
D:\Program.exe  
tomato  
bye  
  
Process returned 0 (0x0)  
Press any key to continue.
```

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

- Ако включим скоби, се изпълнява съответния блок

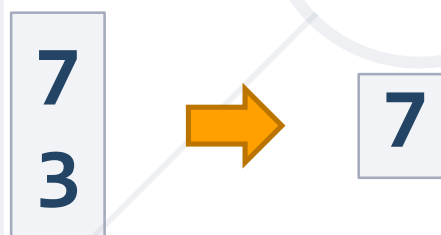
```
string color = "red";  
if (color == "red") {  
    cout << "tomato" << endl;  
    cout << "strawberry" << endl;  
} else if (color == "yellow") {  
    cout << "banana" << endl;  
    cout << "bye" << endl;  
}
```

Изпълняват се редовете
в съответния блок



```
D:\Program.exe  
tomato  
strawberry  
  
Process returned 0 (0x0)  
Press any key to continue.
```

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



Read input

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

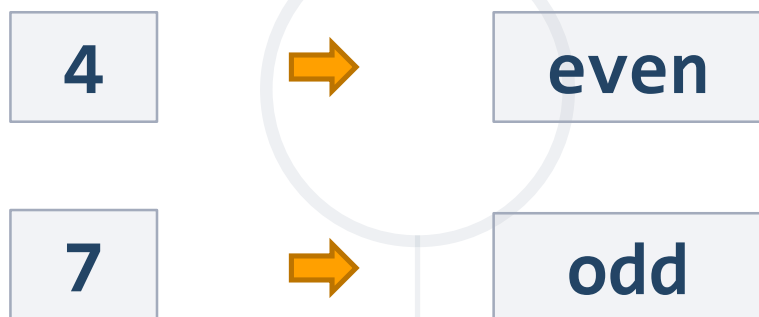
Print output

false

true

Print output

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



Четно или нечетно – решение

```
int num;  
cin >> num;  
if (num % 2 == 0) {  
    cout << "even" << endl;  
}  
else {  
    cout << "odd" << endl;  
}
```




Закръгляне и Форматиране

- В програмирането можем да закръгляме дробни числа

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

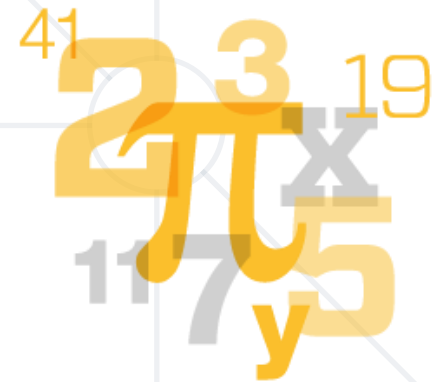
```
double up = ceil(23.45);           // 24.0
```

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

```
double down = floor(45.67);        // 45.0
```

- Намиране на абсолютна стойност

```
int example1 = abs(-50);           // 50  
int example2 = abs(50);            // 50
```



- Фиксиране на изходния поток при извеждане на дробни числа:

```
cout.setf(ios::fixed); // фиксиран формат
```

Задава специфичен
формат на потока

```
cout.precision(2);
```

Брой на цифрите в
дробната част



Дебъгване

Прости операции с дебъгер

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



Breakpoint

```
1  #include <iostream>
2  using namespace std;
3
4  int main()
5  {
6      string currentDay;
7      cin >> currentDay;
8      double myMoney = 0;
9
10     if (currentDay == "31.12.2020")
11     {
12         double salary;
13         cin >> salary;
14         myMoney = myMoney + salary;
15     }
16 }
```

Дебъгване във Visual Studio

- 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 **[F9]**




```
1  #include <iostream>
2  using namespace std;
3
4  int main()
5  {
6      string currentDay;
7      cin >> currentDay;
8      double myMoney = 0;
9
10     if (currentDay == "31.12.2020")
11     {
12         double salary;
13         cin >> salary;
14         myMoney = myMoney + salary;
15     }
16 }
```



Серия от проверки

Серия от проверки

- Конструкцията **if/else-if/else...** може да е в серия




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

TRUE OR **FALSE?**

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

Серия от проверки - пример

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



```
int a = 7;
if (a > 4)
    cout << "Bigger than 4" << endl;
else if (a > 5)
    cout << "Bigger than 5" << endl;
else if (a == 7)
    cout << "Equal to 7" << endl;
```

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

```
string currentDay = "Monday";  
if (currentDay == "Monday") {  
    double salary;  
    cin >> salary;  
}  
cout << salary << endl; // Error!
```

- Напишете програма, която:
 - Прочита **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



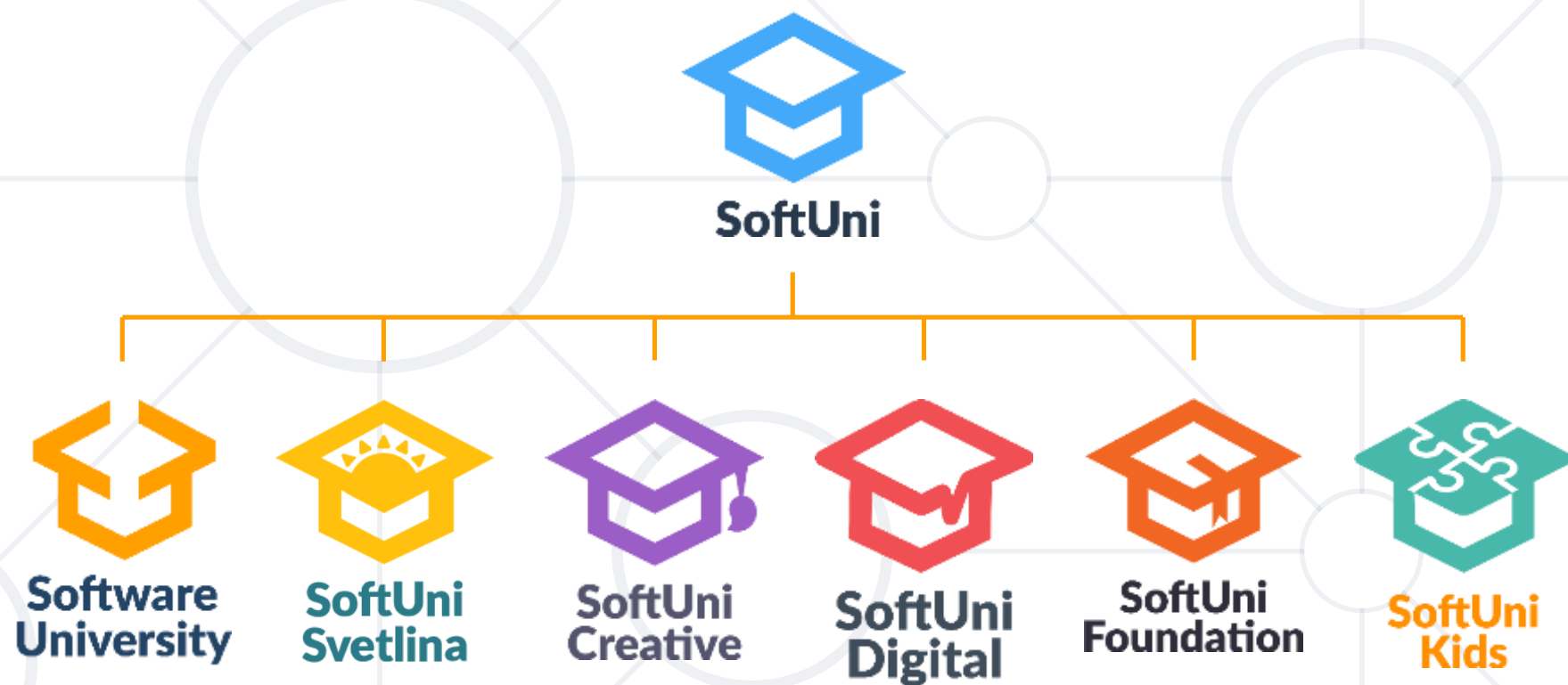
17.5

```
if (figureName == "square"){
    double squareSide;
    cin >> squareSide;
    cout << squareSide * squareSide << endl;
}else if (figureName == "rectangle"){
    double sideA, sideB;
    cin >> sideA >> sideB;
    cout << sideA * sideB << endl;
}else if (...){
    //TODO: Add more conditional statements
}
```

- Оператори за сравнение
- Конструкции за проверка на условие – **if** и **if-else**
- Закръгляне и форматиране
- Серии от проверки
- Дебъгване
- 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>



Обучения в СофтУни

- Software University – High-Quality Education, Profession and Job for Software Developers
 - softuni.bg
- Software University Foundation
 - <http://softuni.foundation/>
- Software University @ Facebook
 - facebook.com/SoftwareUniversity
- Software University Forums
 - forum.softuni.bg

