# Лаб: Условни конструкции

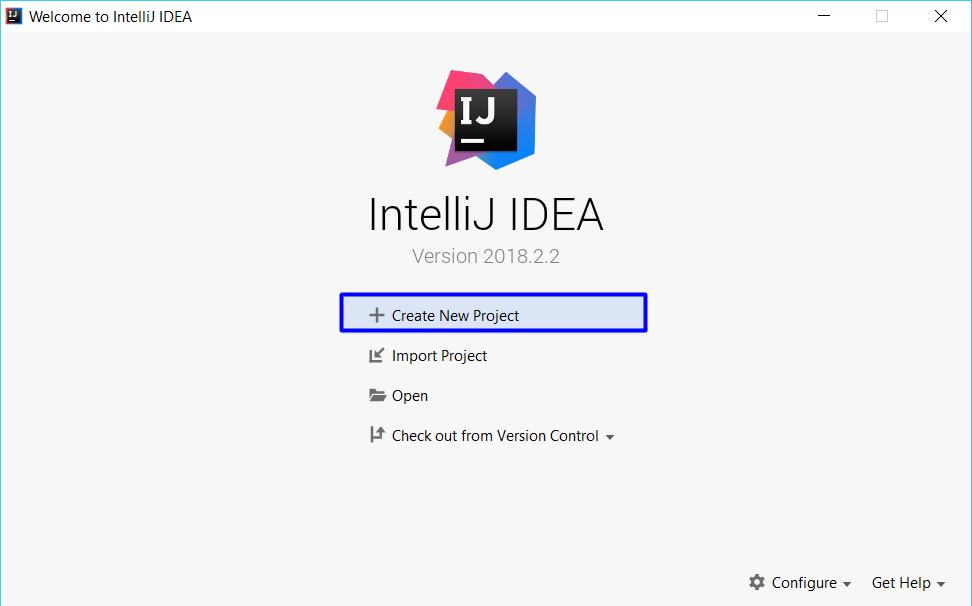
Задачи за упражнение в клас и за домашно към курса ["Основи на програмирането" @ СофтУни](https://softuni.bg/courses/programming-basics).

## Празно IntelliJ решение (Blank Project)

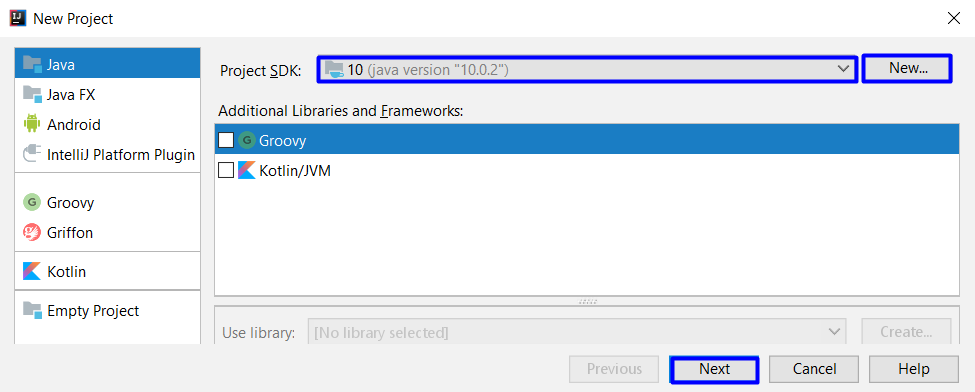
Създайте празно решение (Blank Project) във IntelliJ. Класовете в IntelliJ обединяват група проекти. Тази възможност е изключително удобна, когато искаме да работим по няколко проекта и бързо да превключваме между тях или искаме да обединим логически няколко взаимосвързани проекта.

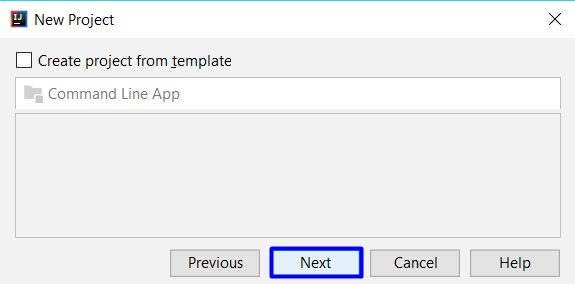
В настоящото практическо занимание ще използваме Java проект и няколко класа, за да организираме решенията на задачите от упражненията - всяка задача в отделен клас в общ проект.

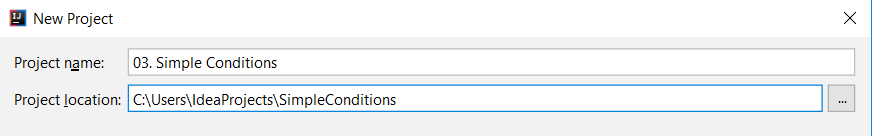
1. Стартирайте IntelliJ IDEA.
2. Създайте нов проект: [Create New Project].



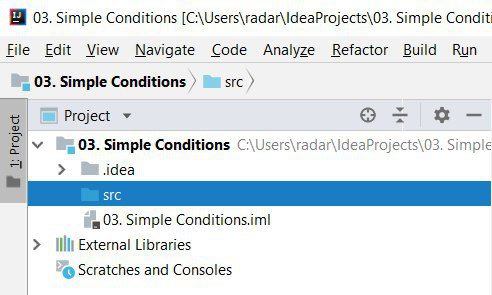
1. Изберете от диалоговия прозорец [Java] 🡪 [Next] 🡪 [Next] и дайте подходящо име на проекта, например "Simple Conditions":







Сега имате създаден **празен IntelliJ проект** (без класове в него):



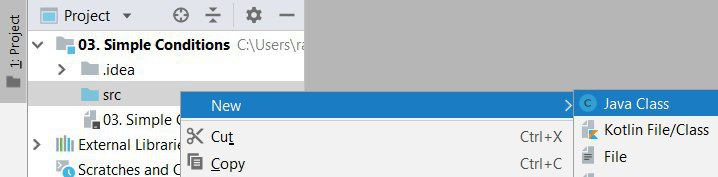
Целта на този **празен проект** e да добавяте в него **по един клас за всяка задача** от упражненията.

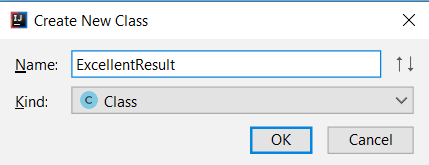
## Проверка за отлична оценка

Първата задача от тази тема е да се напише **конзолна програма**, която **чете оценка** (десетично число), въведена от потребителя, и отпечатва "**Excellent!"** ако оценката е **5.50** или по-висока.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 6 | Excellent! | 5 | *(няма изход)* | 5.50 | Excellent! | 5.49 | *(няма изход)* |

1. Създайте **нов клас** в съществуващото IntelliJ решение. Кликнете с десен бутон на мишката върху **папката "src"**. Изберете [New] 🡪 [Java Class]:



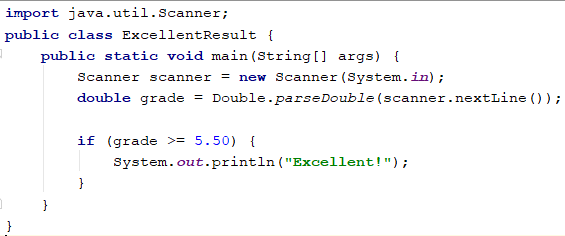


Вече имате проект с един клас в него. Остава да напишете кода за решаване на задачата.

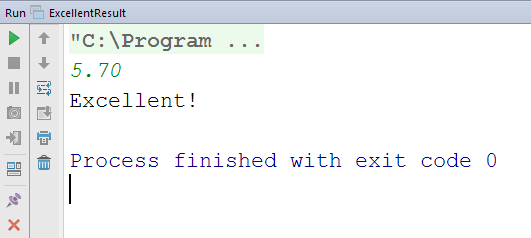
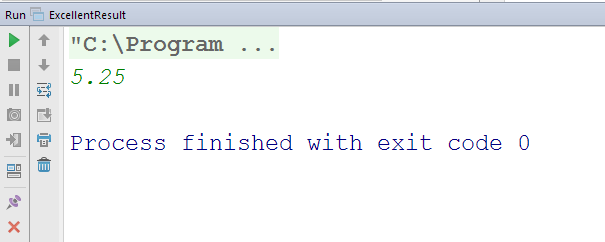
1. Създайте **main метод** като отидете в тялото на класа "ExcellentResult" (между квадратните скоби) и напишете:

|  |
| --- |
| public static void main(String[] args) {  } |

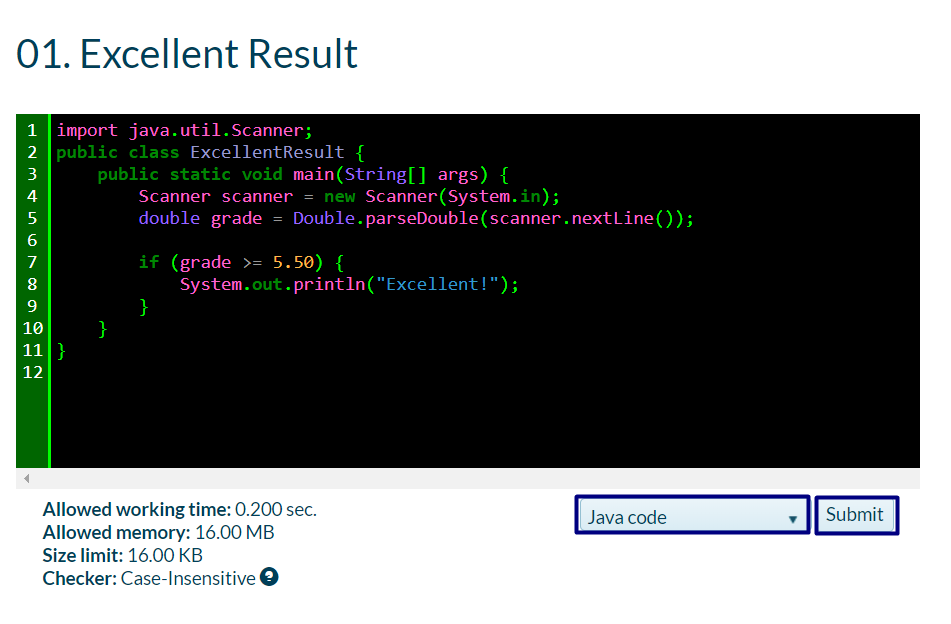
1. Отидете в тялото на метода main(String[] args) (между къдравите скоби) и напишете решението на задачата. Можете да си помогнете с кода от картинката по-долу:



1. **Стартирайте** програмата с [Ctrl+Shift+F10] и я **тествайте** с различни входни стойности:

1. **Тествайте** решението си в **judge системата**: <https://judge.softuni.bg/Contests/Compete/Index/1012#0>
2. Трябва да получите **100 точки** (напълно коректно решение):

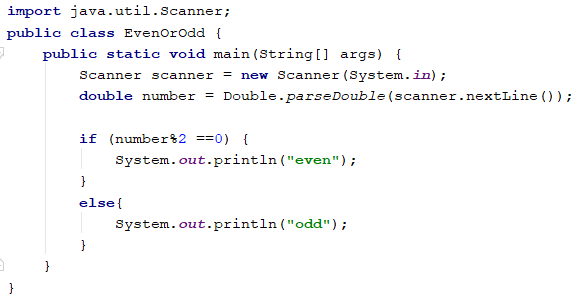


## Четно или нечетно

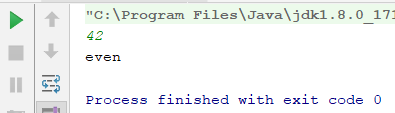
Да се напише програма, която чете **цяло число**, въведено от потребителя, и печата дали е **четно** или **нечетно**. **Примерен вход и изход**

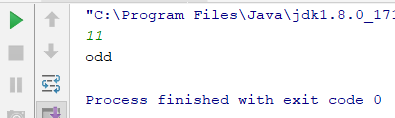
|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 2 | even | 3 | odd | 25 | odd | 1024 | even |

1. Първо добавете **нов Java клас**.
2. **Напишете кода** на програмата. Можете да си помогнете с кода от картинката по-долу:



1. **Стартирайте** програмата с **[Ctrl+Shift+F10]** и я тествайте с различни стойности:





1. Тествайте решението си в **judge системата**: <https://judge.softuni.bg/Contests/Compete/Index/1012#1>

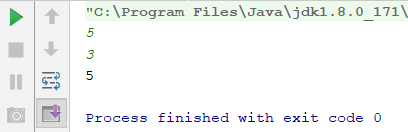
## Намиране на по-голямото число

Да се напише програма, която чете **две цели числа**, въведени от потребителя, и отпечатва **по-голямото от двете**.

**Примерен вход и изход**

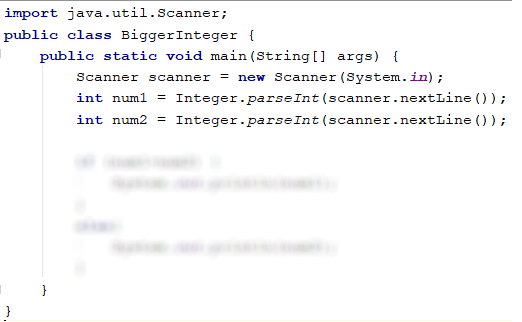
|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 5  3 | 5 | 3  5 | 5 | 10  10 | 10 | -5  5 | 5 |

1. Първо добавете **нов Java клас** в съществуващия solution
2. **Напишете кода** на програмата. Необходима е единична if-else конструкция.
3. **Стартирайте** програмата с **[Ctrl+Shift+F10]** и я тествайте:



1. Тествайте решението си в **judge системата**: <https://judge.softuni.bg/Contests/Compete/Index/1012#2>

**Подсказка**: може да си помогнете частично с кода от картинката, който е нарочно замъглен, за да помислите как да си го напишете сами:



## Изписване на число до 9 с думи

Да се напише програма, която чете **цяло число в диапазона [1…9]**, въведено от потребителя, и го **изписва с думи** на английски език. Ако числото е извън диапазона, изписва "**number too big**".

**Примерен вход и изход**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 5 | five | 1 | one | 9 | nine | 10 | number too big |

Тествайте решението си в **judge системата**: <https://judge.softuni.bg/Contests/Compete/Index/1012#3>

**Подсказка**: можете да напишете дълга if-else-if-else-if…else, с която да разгледате всички възможни случай.

## Познай паролата

Да се напише програма, която **чете парола** (един ред с произволен текст), въведена от потребителя, и проверява дали въведеното **съвпада** с фразата "s3cr3t!P@ssw0rd".При съвпадение да се изведе "**Welcome**". При несъвпадение да се изведе "**Wrong password!**".

**Примерен вход и изход**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| qwerty | Wrong password! | s3cr3t!P@ssw0rd | Welcome | s3cr3t!p@ss | Wrong password! |

Тествайте решението си в **judge системата**: <https://judge.softuni.bg/Contests/Compete/Index/1012#4>

**Подсказка**: използвайте if-else конструкцията.

## Число от 100 до 200

Да се напише програма, която **чете цяло число**, въведено от потребителя, и проверява дали е **под 100**, **между 100 и 200** или **над 200**. Да се отпечатат съответно съобщения като в примерите по-долу:

**Примерен вход и изход**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 95 | Less than 100 | 120 | Between 100 and 200 | 210 | Greater than 200 |

Тествайте решението си в **judge системата**: <https://judge.softuni.bg/Contests/Compete/Index/1012#5>

**Подсказка**: използвайте if-else-if-else конструкция за да проверите всеки от трите случая.

## Еднакви думи

Да се напише програма, която **чете две думи**, въведени от потребителя, и проверява дали са еднакви. Да не се прави разлика между главни и малки думи. Да се изведе "yes" или "no".

**Примерен вход и изход**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| Hello  Hello | yes | SoftUni  softuni | yes | Soft  Uni | no | beer  vodka | no | HeLlO  hELLo | yes |

Тествайте решението си в **judge системата**: <https://judge.softuni.bg/Contests/Compete/Index/1012#6>

**Подсказка**: използвайте if-else конструкция. При сравняване на думите ги обърнете в долен регистър с метода **equalsIgnoreCase**.

## Лица на фигури

Да се напише програма, в която потребителят **въвежда вида и размерите на геометрична** фигура и пресмята лицето й. Фигурите са четири вида: квадрат (**square**), правоъгълник (**rectangle**), кръг (**circle**) и триъгълник (**triangle**). На първия ред на входа се чете вида на фигурата (square, rectangle, circle или triangle). Ако фигурата е **квадрат**, на следващия ред се чете едно число - дължина на страната му. Ако фигурата е **правоъгълник**, на следващите два реда четат две числа - дължините на страните му. Ако фигурата е **кръг**, на следващия ред чете едно число - радиусът на кръга. Ако фигурата е **триъгълник**, на следващите два реда четат две числа - дължината на страната му и дължината на височината към нея. Резултатът да се закръгли до **3 цифри след десетичната точка**.

**Примерен вход и изход**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| square  5 | 25.000 | rectangle  7  2.5 | 17.500 | circle  6 | 113.097 | triangle  4.5  20 | 45.000 |

Тествайте решението си в **judge системата**: <https://judge.softuni.bg/Contests/Compete/Index/1012#7>

**Подсказка**: използвайте серия от if-else-if-else**-**… конструкции, за да обработите 4-те вида фигури.

## Еднакви 3 числа

Да се въведат **3 числа и да се отпечата дали са еднакви** (yes / no).

**Примерен вход и изход**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 1  1  1 | yes | 5  5  5 | yes | 1  2  3 | no | 11  8  5 | no | 13  14  99 | no |

Тествайте решението си в **judge системата**: <https://judge.softuni.bg/Contests/Compete/Index/1012#8>

**Примерна изпитна задача**

## Магазин за детски играчки

*Тествайте решението си* [***тук***](https://judge.softuni.bg/Contests/Compete/Index/1012#9)*.*

Петя има магазин за детски играчки. Тя получава голяма поръчка, която трябва да изпълни. С парите, които ще спечели иска да отиде на екскурзия. Да се напише програма, която пресмята печалбата от поръчката.

**Цени на играчките:**

* **Пъзел - 2.60 лв.**
* **Говореща кукла - 3 лв.**
* **Плюшено мече - 4.10 лв.**
* **Миньон - 8.20 лв.**
* **Камионче - 2 лв.**

Ако поръчаните играчки са **50 или повече** магазинът прави **отстъпка 25%** **от общата цена**. От спечелените пари Петя трябва да даде **10% за наема** на магазина. Да се пресметне дали парите ще ѝ стигнат да отиде на екскурзия.

### Вход

От конзолата се четат **6 реда**:

1. **Цена на екскурзията - реално число в интервала [1.00 … 10000.00]**
2. **Брой пъзели - цяло число в интервала [0… 1000]**
3. **Брой говорещи кукли - цяло число в интервала [0 … 1000]**
4. **Брой плюшени мечета - цяло число в интервала [0 … 1000]**
5. **Брой миньони - цяло число в интервала [0 … 1000]**
6. **Брой камиончета - цяло число в интервала [0 … 1000]**

### Изход

На конзолата се отпечатва:

* Ако **парите са достатъчни** се отпечатва:
  + **"Yes! {оставащите пари} lv left."**
* Ако **парите НЕ са достатъчни** се отпечатва:
  + **"Not enough money! {недостигащите пари} lv needed."**

**Резултатът трябва да се форматира до втория знак след десетичната запетая**.

### Примерен вход и изход

|  |  |  |
| --- | --- | --- |
| **Вход** | **Изход** | **Обяснения** |
| 40.8  20  25  30  50  10 | Yes! 418.20 lv left. | **Сума**: 20 \* 2.60 + 25 \* 3 + 30 \* 4.10 + 50 \* 8.20 + 10 \* 2 = **680** лв.  **Брой на играчките**: 20 + 25 + 30 + 50 + 10 = **135**  **135 > 50 => 25% отстъпка**; 25% от 680 = **170 лв. отстъпка**  **Крайна цена**: 680 – 170 = **510** лв.  **Наем**: 10% от 510 лв. = **51** лв.  **Печалба**: 510 – 51 = **459** лв.  **459 > 40.8** =>459 – 40.8= **418.20** лв. **остават** |
| **Вход** | **Изход** | **Обяснения** |
| 320  8  2  5  5  1 | Not enough money! 238.73 lv needed. | **Сума**: **90.3** лв.  **Брой на играчките**: **21**  **21 < 50 => няма отстъпка**  **Наем**: 10% от 90.3 = **9.03** лв.  **Печалба**: 90.3 – 9.03 = **81.27** лв.  **81.27 < 320** => 320 – 81.27= **238.73** лв. **не достигат** |