# Лаб: Условни конструкции

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

Тествайте решенията си в **Judge** системата: [https://judge.softuni.bg/Contests/Compete/Index/1012](https://judge.softuni.bg/Contests/Compete/Index/1012#0)

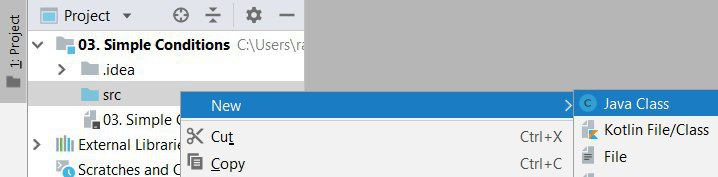
## Проверка за отлична оценка

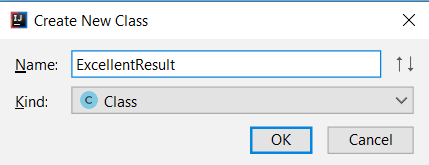
Първата задача от тази тема е да се напише **конзолна програма**, която **чете оценка** (десетично число), въведена от потребителя, и отпечатва "**Excellent!"** ако оценката е **5.50** или по-висока.

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

### Насоки:

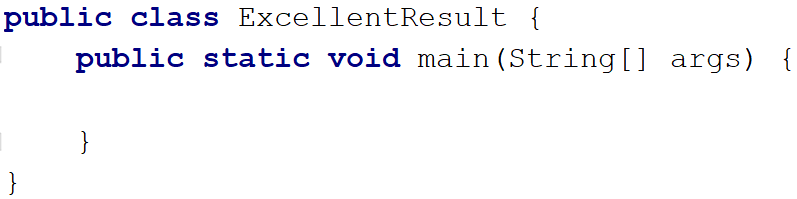
1. Създайте **нов клас** в съществуващия IntelliJ проект. Кликнете с десен бутон на мишката върху **папката "**src**"**. Изберете [New] 🡪 [Java Class]:



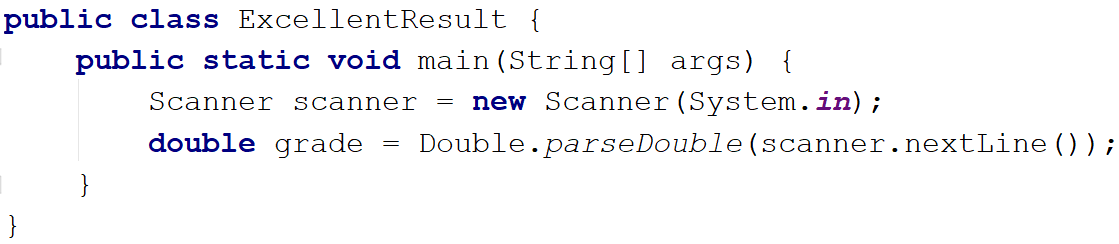


Вече имате проект с един клас в него. Остава да напишете кода за решаване на задачата.

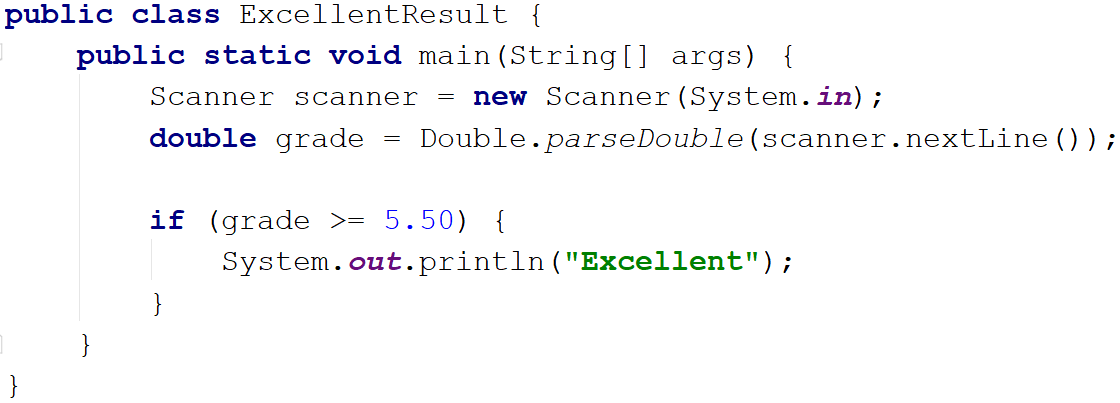
1. Създайте main **метод** като отидете в класа "ExcellentResult" (между квадратните скоби) и напишете:



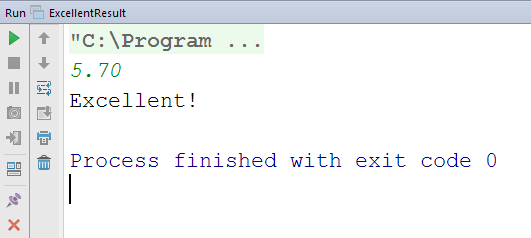
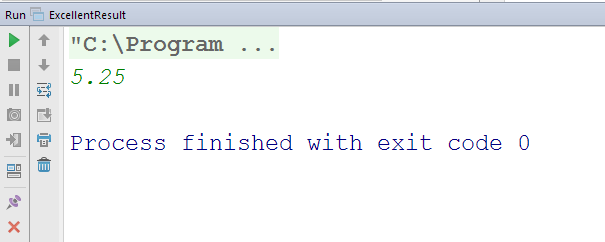
1. Отидете в тялото на метода main(String[] args) (между къдравите скоби). Създайте Scanner обект, с който да четете от конзолата и прочетете едно **реално** **число** - оценката:



1. Направете проверка за стойността на оценката. Ако тя е по-голяма или равна на 5.50, отпечатайте изхода по условие:



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

## Намиране на по-голямото число

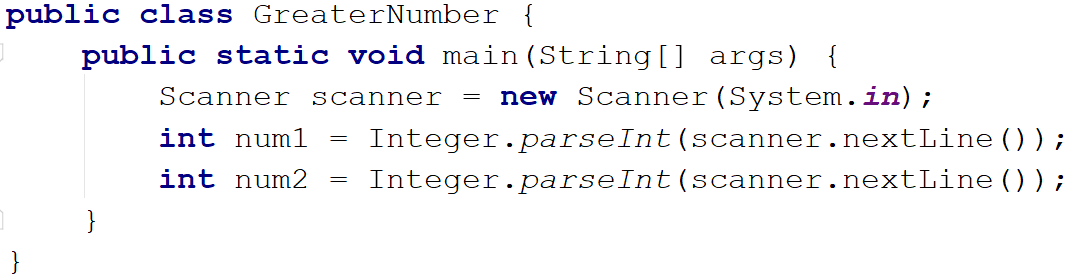
Да се напише програма, която чете **две цели числа**, въведени от потребителя, и отпечатва **по-голямото от двете**.

**Примерен вход и изход**

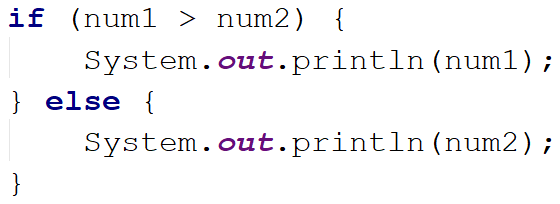
|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 5  3 | 5 | 3  5 | 5 | 10  10 | 10 | -5  5 | 5 |

### Насоки:

1. Прочетете 2 цели числа от конзолата:



1. Сравнете дали първото число num1 e по-голямо от второто num2. Отпечатайте по-голямото число.



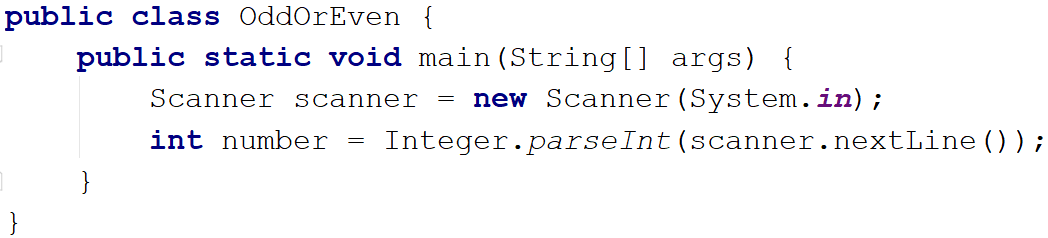
## Четно или нечетно

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

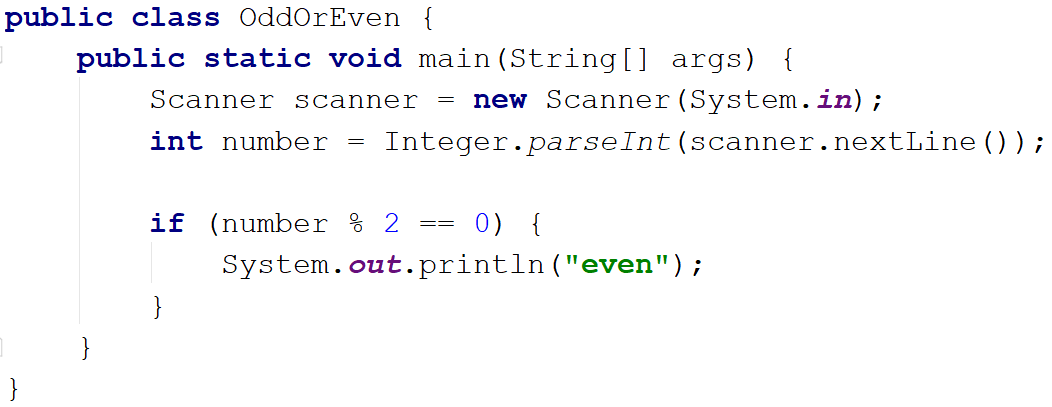
|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 2 | even | 3 | odd | 25 | odd | 1024 | even |

### Насоки:

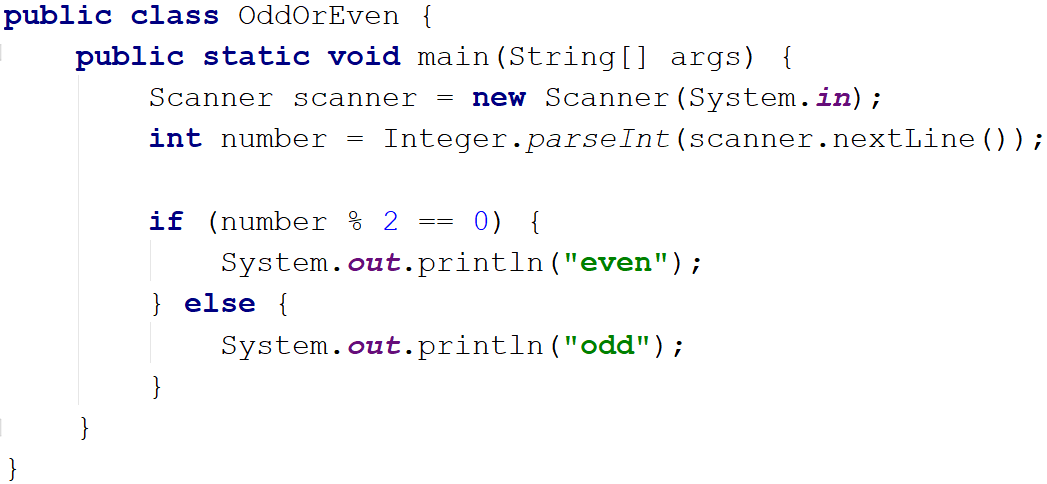
1. Първо добавете **нов Java клас** към съществуващия проект
2. Създайте Scanner обект и прочетете едно цяло число от конзолата:



1. Проверете дали числото е четно като го разделите на 2 и проверите дали има остатък от делението. Отпечатайте изхода по условие – текста “even”:



1. В противен случай отпечатайте “odd”:



## Изписване на число до 9 с думи

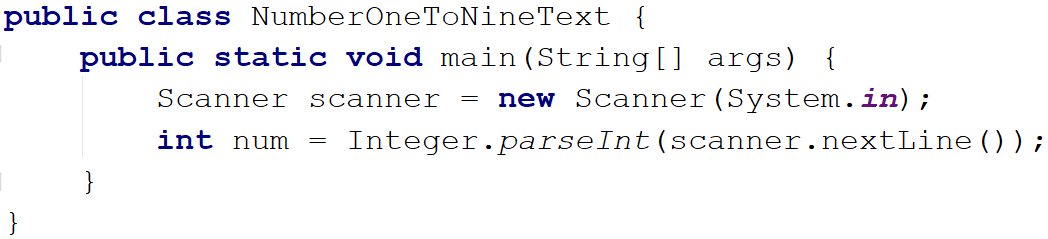
Да се напише програма, която чете **цяло число в диапазона [1…9]**, въведено от потребителя, и го **изписва с думи** на английски език. Ако числото е извън диапазона, изписва "**number too big**".

**Примерен вход и изход**

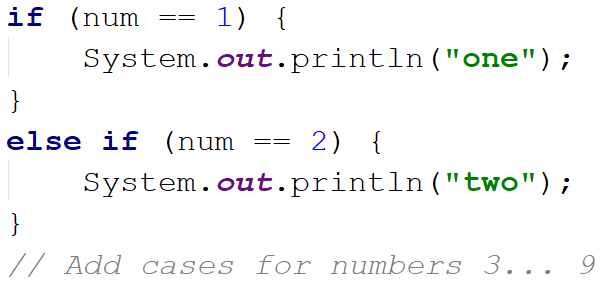
|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 5 | five | 1 | one | 9 | nine | 10 | number too big |

### Насоки

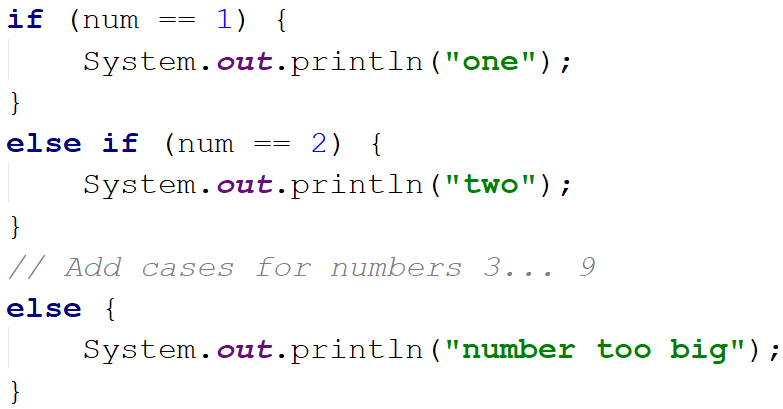
1. Прочетете едно цяло число от конзолата:



1. Направете проверка за размера на числото. Принтирайте с текст някоя от думите [“one”, “two”, “three”… “nine”] в зависимост от стойността.



1. Отпечатайте “number too big”, ако числото е по-голямо от 9:



## 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 |

## Число от 100 до 200

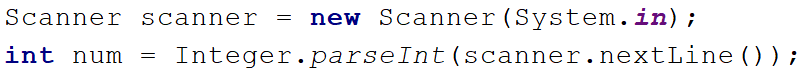
Да се напише програма, която **чете цяло число**, въведено от потребителя, и проверява дали е **под 100**, **между 100 и 200** или **над 200**. Да се отпечатат съответно съобщения като в примерите по-долу:

**Примерен вход и изход**

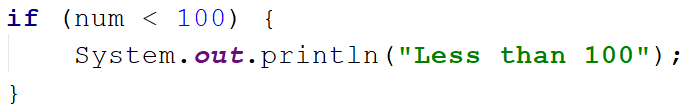
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 95 | Less than 100 | 120 | Between 100 and 200 | 210 | Greater than 200 |

### Насоки:

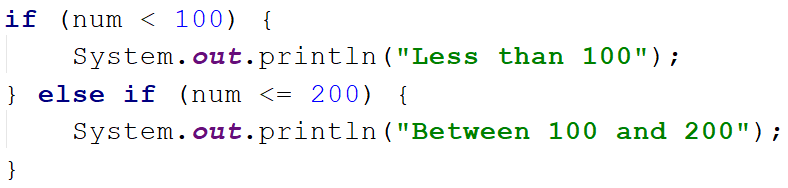
1. Прочетете едно цяло число от конзолата:



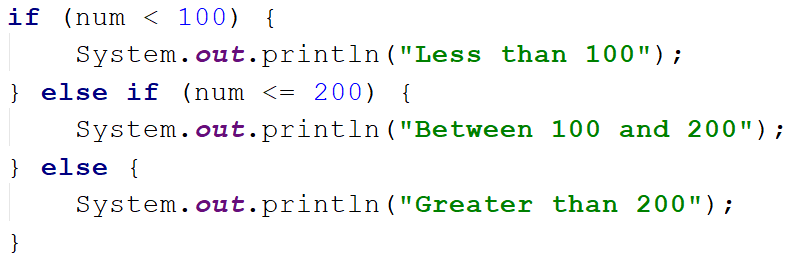
1. Проверете дали числото е **по-малко от 100**:



1. Ако числото не е по-малко от 100, проверете дали е **по-малко или равно на 200**:



1. Ако числото не е по-малко от 200, то то е по-голямо. Изпишете текста по условие:



## Познай паролата

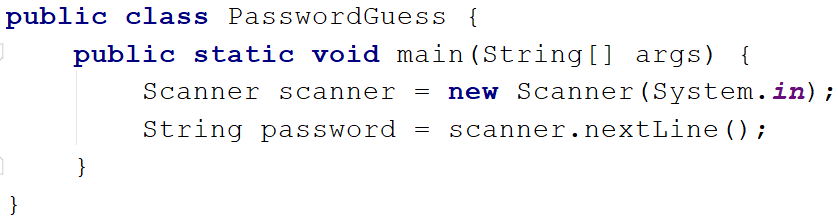
Да се напише програма, която **чете парола** (един ред с произволен текст), въведена от потребителя, и проверява дали въведеното **съвпада** с фразата "s3cr3t!P@ssw0rd".При съвпадение да се изведе "**Welcome**". При несъвпадение да се изведе "**Wrong password!**".

**Примерен вход и изход**

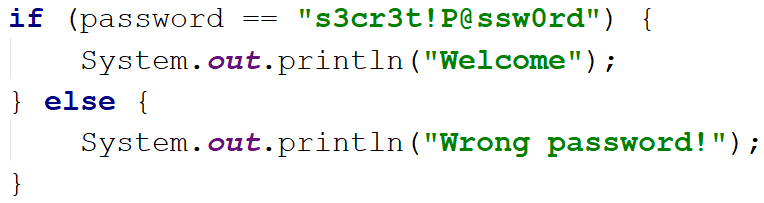
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| qwerty | Wrong password! | s3cr3t!P@ssw0rd | Welcome | s3cr3t!p@ss | Wrong password! |

### Насоки:

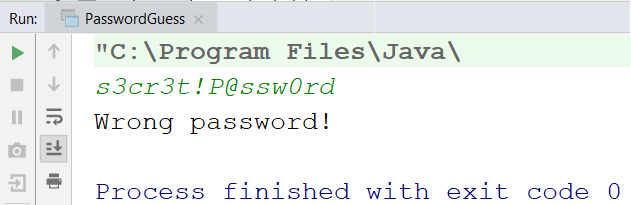
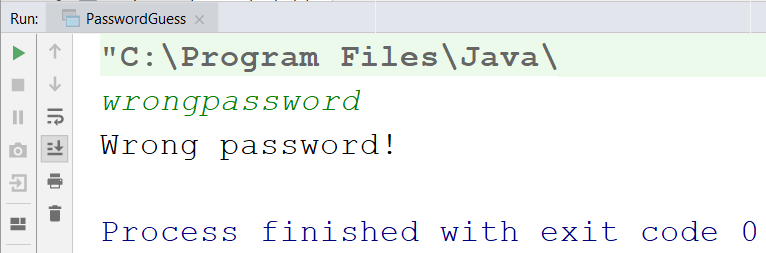
1. Прочетете входните данни – eдин ред текст от конзолата:



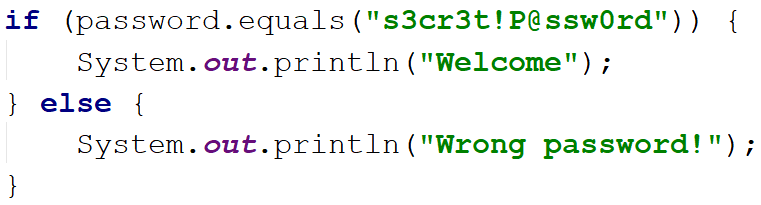
1. Сравнете стойността на прочетената парола с текста “s3cr3t!P@ssw0rd ”. Отпечатайте правилния изход, спрямо въведената парола:



Тествайте написаната програмна логика до момента:



Програмата не работи коректно при въвеждане на правилна парола. Защо се получава това? Операторът “==” не работи така, както предполагаме за текстови стойности. Той сравнява текстовата стойност по адрес в паметта. В този смисъл, променливата password и текстът “s3cr3t!P@ssw0rd” имат различен адрес в паметта на компютъра. Вместо оператора “==” използвайте метода .equals():



Прочетете повече за сравняването с типа string [тук](https://www.baeldung.com/java-compare-strings).

## Еднакви думи

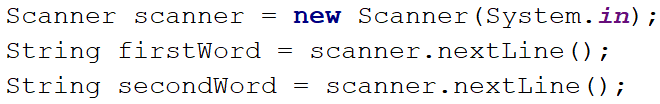
Да се напише програма, която **чете две думи**, въведени от потребителя, и проверява дали са еднакви. Да не се прави разлика между главни и малки думи. Да се изведе "yes" или "no".

**Примерен вход и изход**

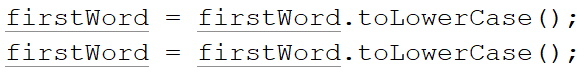
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| Hello  Hello | yes | SoftUni  softuni |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

### Насоки:

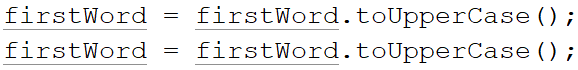
1. Прочетете 2 думи от конзолата:



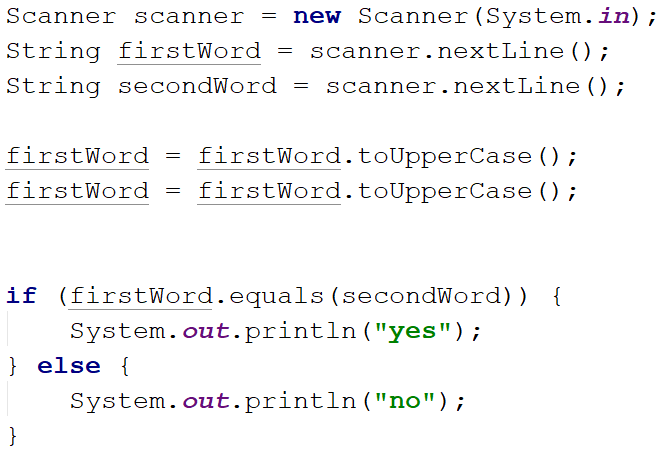
1. След като сте прочели думите, направете всички букви от първата и втората дума да бъдат малки, използвайки метода toLowerCase(). Можете да направите и обратната операция, използвайки метода toUpperCase():



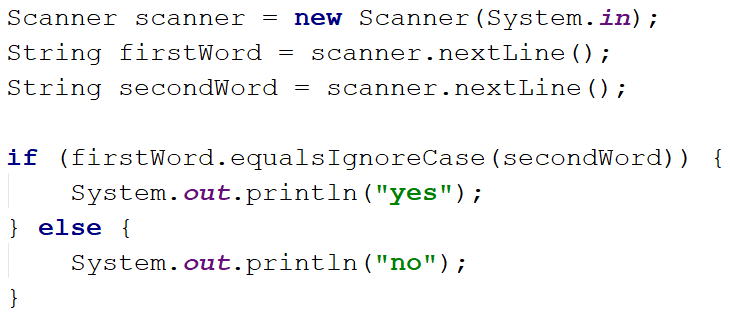
или:



1. Сравнете двете думи, след като сте уеднаквили по някой от горепосочените начини всички букви и принтирайте изхода по условие:



Алтернативно можете да използвате метода equalsIgnoreCase(), който сравнява текстови стойности без значение буквите в тях:



## Лица на фигури

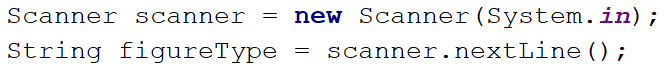
Да се напише програма, в която потребителят **въвежда вида и размерите на геометрична** фигура и пресмята лицето й. Фигурите са четири вида: квадрат (**square**), правоъгълник (**rectangle**), кръг (**circle**) и триъгълник (**triangle**). На първия ред на входа се чете вида на фигурата (square, rectangle, 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**

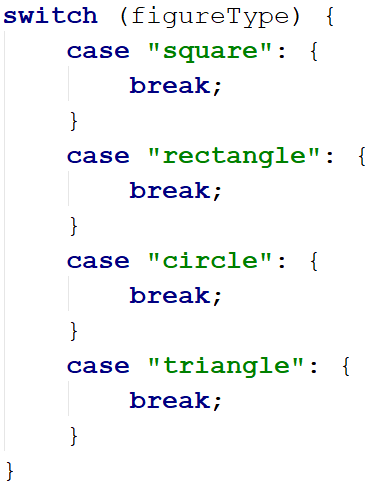
|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| square  5 | 25.000 | rectangle  7  2.5 | 17.500 | circle  6 | 113.097 | triangle  4.5  20 | 45.000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

### Насоки

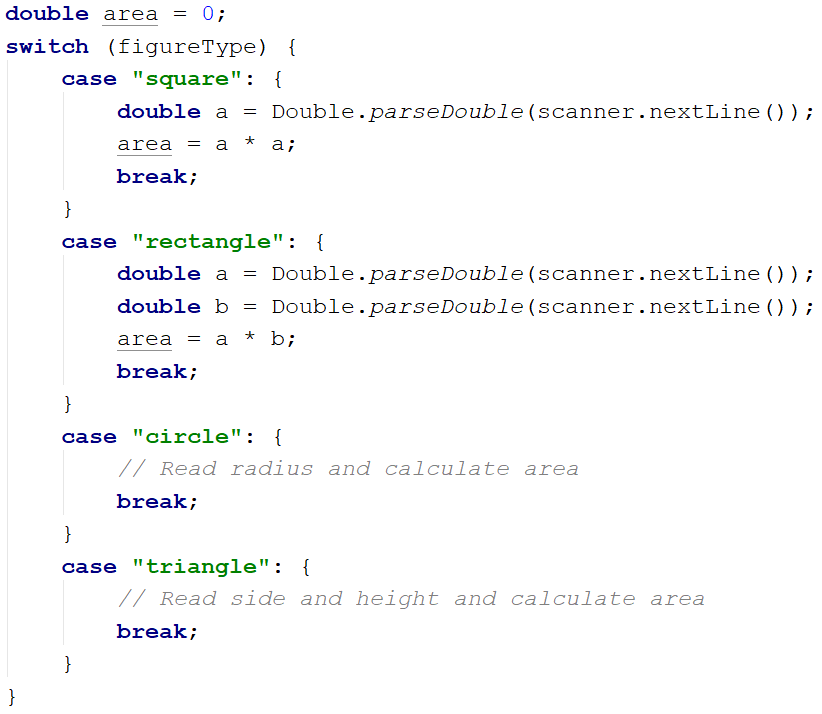
1. Прочетете тип на фигурата – текст въведен от конзолата:



1. Направете switch конструкция, която съдържа случаи за типа фигури:



1. Във всеки случай, четете нужния брой страни и извършвайте математически операции за намиране на лицето според типа на фигурата. Направете променлива area, в която да пазите пресметнатото лице на фигурата:



1. Принтирайте стойността на променливата area, като форматирате стойността й до третия знак след десетичната запетая:



## Ден от седмицата

Напишете програма, която чете **цяло** **число**, въведено от потребителя, и отпечатва **ден от седмицата** (на английски език), в граници [1...7] или отпечатва "Error" в случай, че въведеното число е **невалидно**.

### Примерен вход и изход

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 1 | Monday |
| 2 | Tuesday |
| 3 | Wednesday |
| 4 | Thursday |
| 5 | Friday |
| 6 | Saturday |
| 7 | Sunday |
| -1 | Error |

**\*Подсказка:** Използвайте условната конструкция [**switch-case**](https://docs.oracle.com/javase/tutorial/java/nutsandbolts/switch.html).

## Клас животно

Напишете програма, която отпечатва класа на животното според неговото име, въведено от потребителя.

* **dog -> mammal**
* **crocodile, tortoise, snake -> reptile**
* **others -> unknown**

### Примерен вход и изход

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| dog | mammal |
| snake | reptile |
| cat | unknown |

**\*Подсказка:** Използвайте условната конструкция switch-case.

**Примерна изпитна задача**

## Магазин за детски играчки

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."**

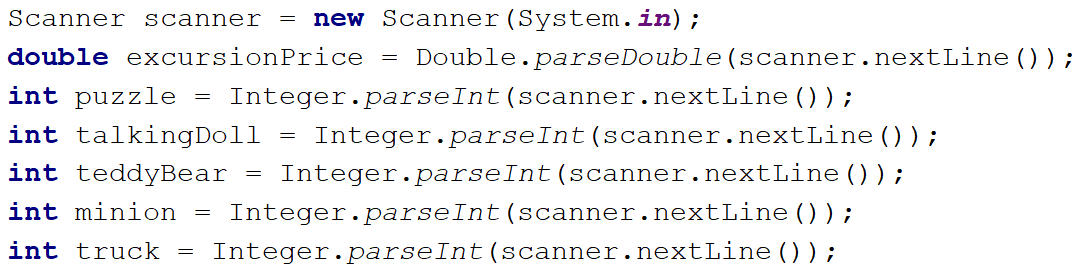
**Резултатът трябва да се форматира до втория знак след десетичната запетая**.

### Примерен вход и изход

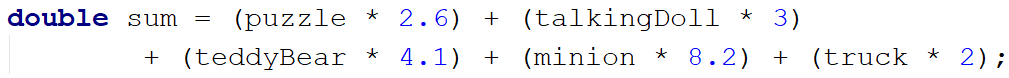
|  |  |  |
| --- | --- | --- |
| **Вход** | **Изход** | **Обяснения** |
| 40.8  20  25  30  50  10 | Yes! 418.20 lv left. | **Сума**: 20 \* 2.60 + 25 \* 3 + 30 \* 4.10 + 50 \* 8.20 + 10 \* 2 = **680** лв.  **Брой на играчките**: 20 + 25 + 30 + 50 + 10 = **135**  **135 > 50 => 25% отстъпка**; 25% от 680 = **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** |

### Насоки:

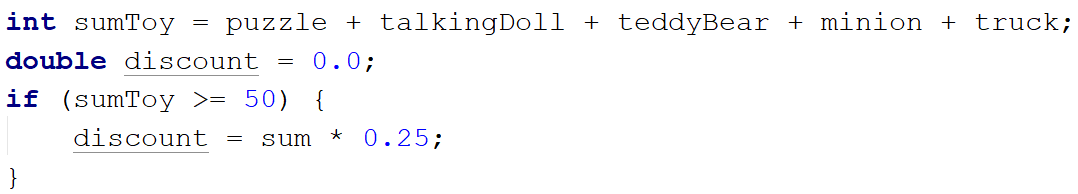
1. Прочетете входните данни – броя на играчките и цената на екскурзията:



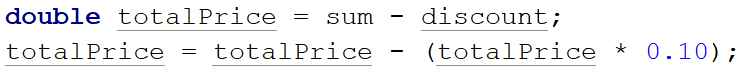
1. Пресметнете общата цена на играчките, като умножите броя на всеки тип играчки по единичната им цена:



1. Изчислете общия брой на играчките. Създайте променлива discount, в която да пазите стойността на отстъпката. Направете проверка дали броя на играчките надвишава 50 и ако е вярно, изчислете отстъпката и присвоете стойността към променливата discount:



1. Изчислете крайната цена, като извадите от сумата за всички играчки отстъпката, която сте изчислили, както и парите за наема:



1. Сравнете крайната печалба с цената на екскурзията. Изпишете правилния изход спрямо резултата:

