# Упражнения: Прости проверки

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.

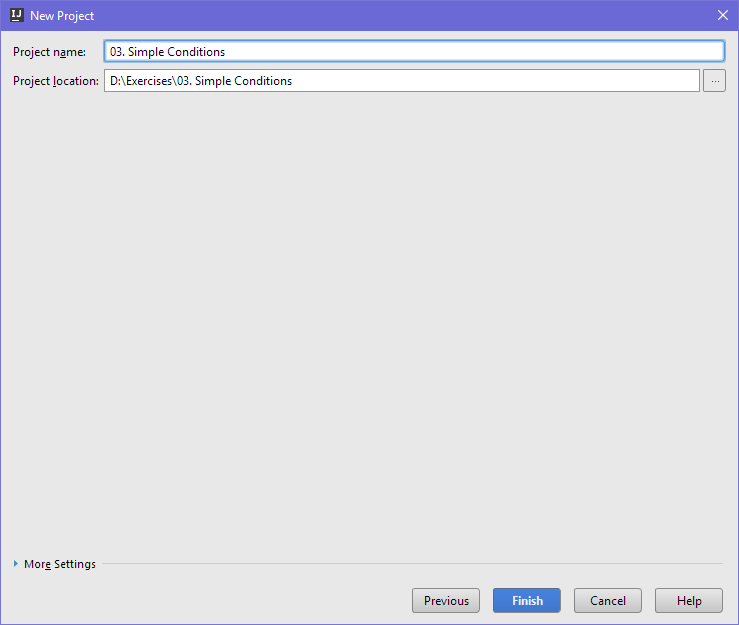
1. Стартирайте IntelliJ IDEA.
2. Създайте нов проект: [Create New Project].



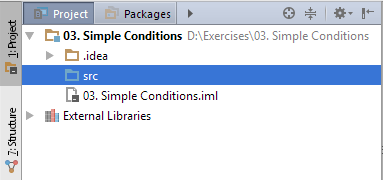
1. Изберете от диалоговия прозорец [Java] 🡪 [Next] 🡪 [Next] и дайте подходящо име на проекта, например “Simple Conditions”:

🡺





Сега имате създаден **празен IntelliJ проект** (с 0 класове в него):



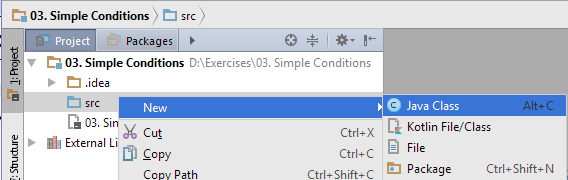
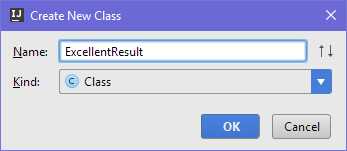
Целта на този **blank project** e да добавяте в него **по един клас за всяка задача** от упражненията.

## 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]:

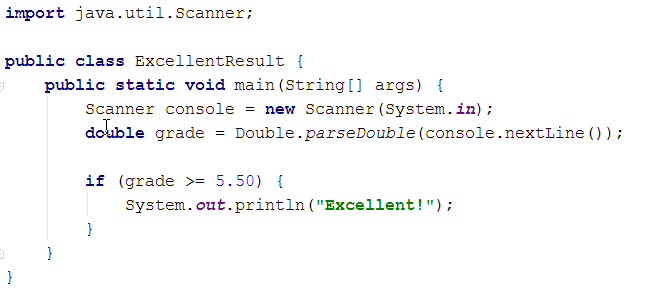
🡺 

Вече имате проект с едно конзолно приложение в него. Остава да напишете кода за решаване на задачата.

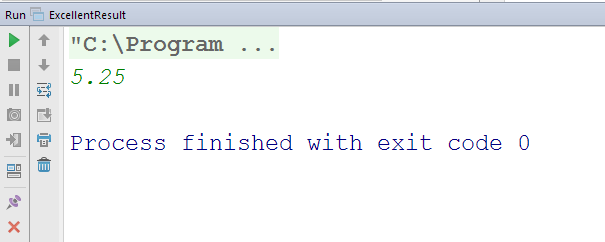
1. Създайте main метод като отидете в тялото на класа „ExcellentResult” (между квадратните скоби) и напишете:

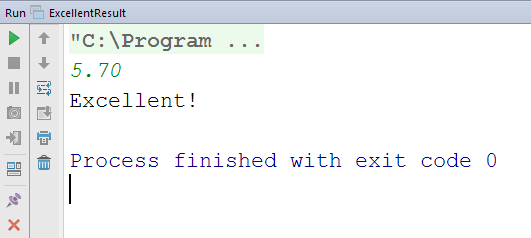
|  |
| --- |
| public static void main(String[] args) {  } |

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

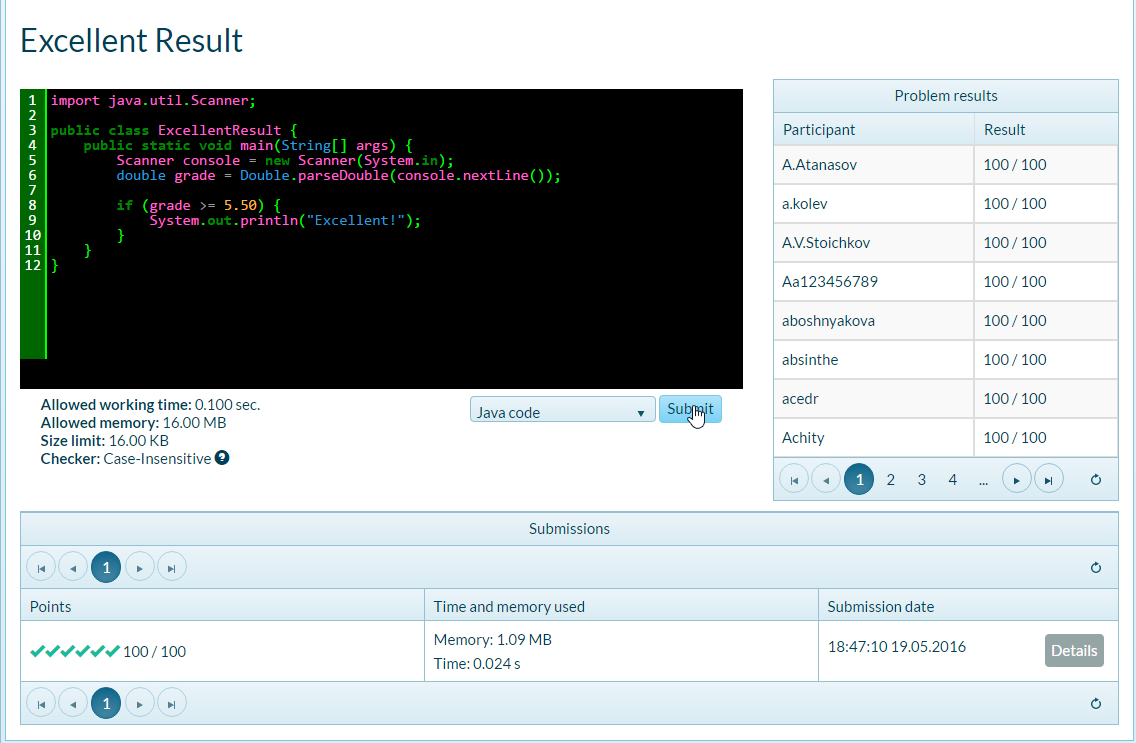


1. **Стартирайте** програмата с [Ctrl+Shift+F10] и я **тествайте** с различни входни стойности:





1. **Тествайте** решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/152#0>. Трябва да получите **100 точки** (напълно коректно решение):

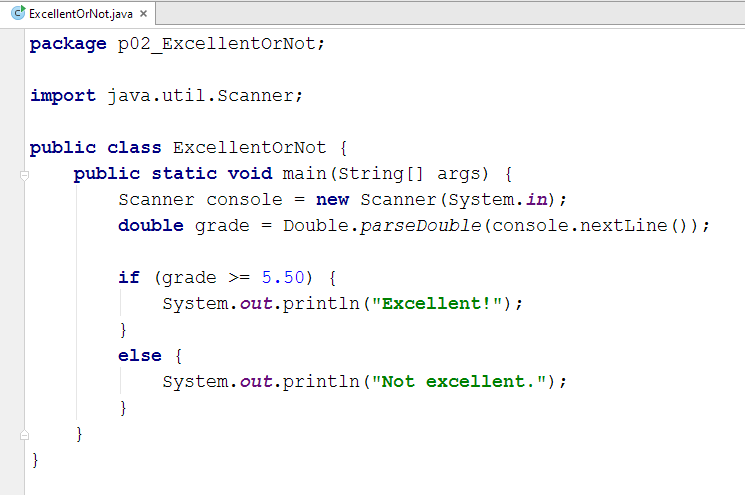


## Отлична оценка или не

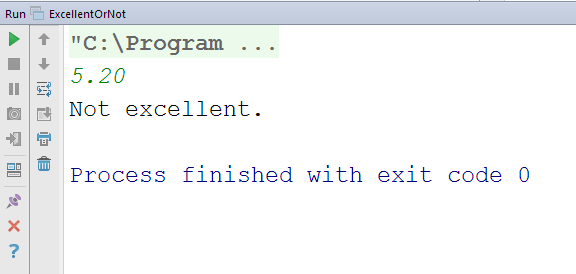
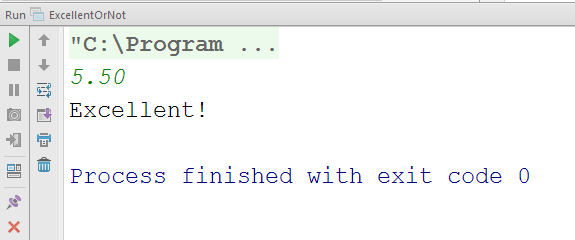
Следващата задача от тази тема е да се напише **конзолна програма**, която **чете оценка** (десетично число), въведена от потребителя, и отпечатва “**Excellent!**”, ако оценката е **5.50** или по-висока, или “**Not excellent.**” в противен случай.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 6 | Excellent! | 5 | Not excellent. | 5.50 | Excellent! | 5.49 | Not excellent. |

1. Първо създайте **нова Java програма** в решението “**03. Complex Conditions**”.
   * Кликнете с мишката върху папката “src” и създайте в папката нов Java клас от [New] 🡪 [Java Class] с име „ExcellentOrNot”.
2. **Напишете кода** на програмата. Може да си помогнете с примерния код от картинката:



1. Сега **стартирайте програмата**, както обикновено с [Ctrl+Shift+F10] и я тествайте:



1. Тествайте в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/152#1>. Решението би трябвало да бъде прието като напълно коректно:

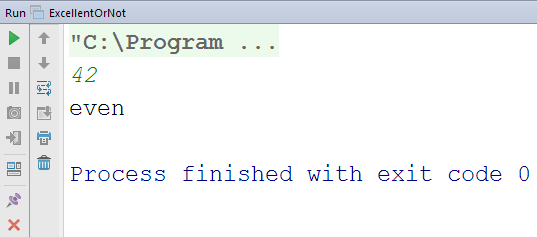


## Четно или нечетно

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

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

1. Първо добавете **нова Java конзолна програма**.
2. **Напишете кода** на програмата. Проверката за четност може да се реализира чрез проверка на **остатъка при деление на 2** по следния начин: boolean even = (num % 2 == 0).
3. **Стартирайте** програмата с **[Ctrl+Shift+F10]** и я тествайте:



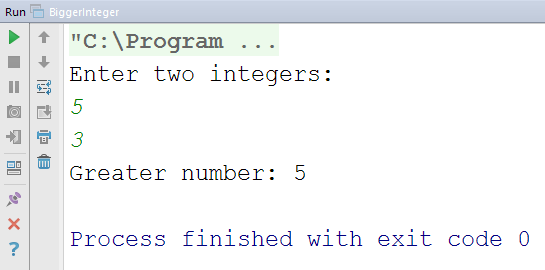
1. Тествайте в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/152#2>.

## Намиране на по-голямото число

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

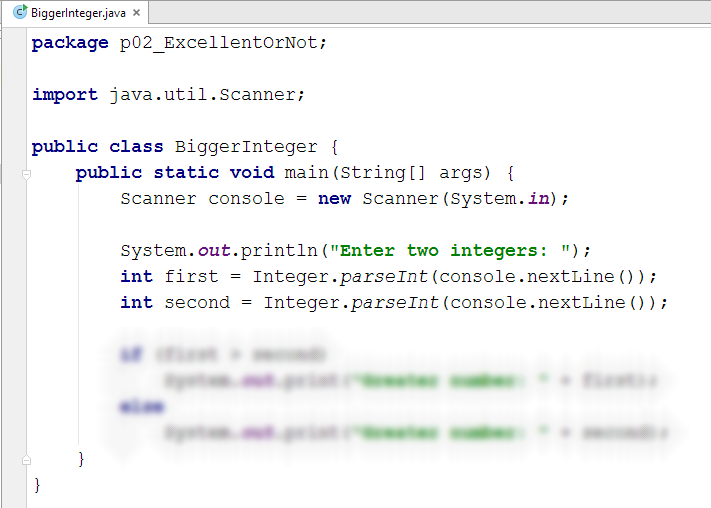
|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 5  3 | 5 | 3  5 | 5 | 10  10 | 10 | -5  5 | 5 |

1. Първо добавете **нов Java конзолен проект** 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/Practice/Index/152#3>.

**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 от 1 до 9 с думи

Да се напише програма, която чете **цяло число в диапазона [0…10]**, въведено от потребителя, и го **изписва с думи** на английски език. Ако числото е извън диапазона, изписва “**number too big**”. Примери:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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/Practice/Index/152#4>.

**Подсказка**: можете да напишете дълга if-else-if-else…else, с която да разгледате възможните **11 случая**.

## Бонус точки

Дадено е **цяло число** – брой точки. Върху него се начисляват **бонус точки** по правилата, описани по-долу. Да се напише програма, която пресмята **бонус точките** за това число и **общия брой точки** с бонусите.

* Ако числото е **до 100** включително, бонус точките са **5**.
* Ако числото е **по-голямо от 100**, бонус точките са **20%** от числото.
* Ако числото е **по-голямо от 1000**, бонус точките са **10%** от числото.
* Допълнителни бонус точки (начисляват се отделно от предходните):
  + За **четно** число 🡪 + 1 т.
  + За число, което **завършва на 5** 🡪 + 2 т.

Примери:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 20 | 6  26 | 175 | 37  212 | 2703 | 270.3  2973.3 | 15875 | 1589.5  17464.5 |

Ето как би могло да изглежда решението на задачата в действие:



Тествайте решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/152#5>.

**Подсказка**:

* Основните бонус точки можете да изчислите с if-else-if-else-if конструкция (имате 3 случая).
* Допълнителните бонус точки можете да изчислите с if-else-if конструкция (имате още 2 случая).

## Сумиране на секунди

Трима спортни състезатели финишират за някакъв **брой секунди** (между **1** и **50**). Да се напише програма, която чете времената на състезателите, въведени от потребителя, и пресмята **сумарното им време** във формат "минути:секунди". Секундите да се изведат с **водеща нула** (2 🡪 "02", 7 🡪 "07", 35 🡪 "35"). Примери:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 35  45  44 | 2:04 | 22  7  34 | 1:03 | 50  50  49 | 2:29 | 14  12  10 | 0:36 |

Тествайте решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/152#6>.

**Подсказка**:

* Сумирайте трите числа и получете резултата в секунди. Понеже **1 минута = 60 секунди**, ще трябва да изчислите броя минути и броя секунди в диапазона от 0 до 59.
* Ако резултатът е между 0 и 59, отпечатайте 0 минути + изчислените секунди.
* Ако резултатът е между 60 и 119, отпечатайте 1 минута + изчислените секунди минус 60.
* Ако резултатът е между 120 и 179, отпечатайте 2 минути + изчислените секунди минус 120.
* Ако секундите са по-малко от 10, изведете водеща нула преди тях.

## Конвертор за мерни единици

Да се напише програма, която **преобразува разстояние** между следните 8 **мерни единици**: m, mm, cm, mi, in, km, ft, yd. Използвайте съответствията от таблицата по-долу:

|  |  |
| --- | --- |
| **входна единица** | **изходна единица** |
| **1** meter (**m**) | **1000** millimeters (**mm**) |
| **1** meter (**m**) | **100** centimeters (**cm**) |
| **1** meter (**m**) | **0.000621371192** miles (**mi**) |
| **1** meter (**m**) | **39.3700787** inches (**in**) |
| **1** meter (**m**) | **0.001** kilometers (**km**) |
| **1** meter (**m**) | **3.2808399** feet (**ft**) |
| **1** meter (**m**) | **1.0936133** yards (**yd**) |

Входните данни се състоят от три реда, въведени от потребителя:

* Първи ред: число за преобразуване
* Втори ред: входна мерна единица
* Трети ред: изходна мерна единица (за резултата)

Резултатът **да се форматира** до **осмия знак след десетичната запетая**.

Примерен вход и изход:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 12  km  ft | 39370. 0788 | 150 mi in | 9503999.99393599 | 450  yd  km | 0.41148000 |

Тествайте решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/152#7>.

## 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!**”. Примери:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **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/Practice/Index/152#8>.

**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:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **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/Practice/Index/152#9>.

**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”. Примери:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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/Practice/Index/152#10>.

**Подсказка**: използвайте if-else конструкция. Преди сравняване на думите ги обърнете в долен регистър: word = word.toLower().

## Информация за скоростта

Да се напише програма, която **чете скорост** (десетично число), въведена от потребителя,и отпечатва **информация за скоростта**. При скорост **до 10** (включително) отпечатайте “**slow**”. При скорост **над 10** и **до 50** отпечатайте “**average**”. При скорост **над 50** и **до 150** отпечатайте “**fast**”. При скорост **над 150** и **до 1000** отпечатайте “**ultra fast**”. При по-висока скорост отпечатайте “**extremely fast**”. Примери:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 8 | slow | 49.5 | average | 126 | fast | 160 | ultra fast | 3500 | extremely fast |

Тествайте решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/152#11>.

**Подсказка**: използвайте серия от if-else-if-else**-**… конструкции, за да хванете всичките 5 случая.

## 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**. Примери:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| square  5 | 25 | rectangle  7  2.5 | 17.5 | circle  6 | 113.097 | triangle  4.5  20 | 45 |

Тествайте решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/152#12>.

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

## Време + 15 минути

Да се напише програма, която **чете час и минути** от 24-часово денонощие, въведени от потребителя, и изчислява колко ще е **часът след 15 минути**. Резултатът да се отпечата във формат hh:mm. Часовете винаги са между 0 и 23, а минутите винаги са между 0 и 59. Часовете се изписват с една или две цифри. Минутите се изписват винаги с по две цифри, с **водеща нула** когато е необходимо. Примери:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 1  46 | 2:01 | 0  01 | 0:16 | 23  59 | 0:14 | 11  08 | 11:23 | 12  49 | 13:04 |

Тествайте решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/152#13>.

**Подсказка**: добавете 15 минути и направете няколко проверки. Ако минутите надвишат 59, увеличете часовете с 1 и намалете минутите със 60. По аналогичен начин разгледайте случая, когато часовете надвишат 23. При печатането на минутите проверете за водеща нула.

## Еднакви 3 числа

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

Примери:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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/Practice/Index/152#14>.

## Изписване на число до 100 с думи

Да се напише програма, която превръща число [0…100] в текст: 25 🡪 “twenty five”. Ако числото не е в диапазона, принтирайте „**invalid number**”.

Примери:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 0 | zero | 67 | sixty seven | -1 | invalid number | 14 | fourteen |

Тествайте решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/152#15>.

# Изпитни задачи от минали издания на курса

## \* Трима братя

*Втора задача от изпитa на 17 септември 2017. Тествайте решението си* [***тук***](https://judge.softuni.bg/Contests/Practice/Index/784#1)*.*

Трима братя решили да изненадат баща си, като почистят заедно гаража му. Големият брат чисти сам за определени часове – **А**. Средният брат чисти сам за **В** часа. По-малкият брат чисти сам за **С** часа. Бащата отива да лови риба и няма да го има **D** часа.

Напишете програма, която пресмята **дали тримата братя могат да изчистят заедно гаража и да изненадат баща си или не.**

Към общото време за почистване се **добавят 15% за почивка.**

### Вход

Входът се **чете от конзолата** и се състои от **4 реда**:

* **Времето на първият брат за чистене сам** – **реално число** в интервала **[0.00 … 99.00]**
* **Времето на вторият брат за чистене сам** – **реално число** в интервала **[0.00 … 99.00]**
* **Времето на третият брат за чистене сам** – **реално число** в интервала **[0.00 …99.00]**
* **Времето за риболов на бащата**– **реално число** в интервала **[0.00 … 99.00]**

### Изход

На конзолата трябва да се **отпечата два реда**.

1. Времето за чистене и почивка, форматирано до втори знак: **“Cleaning time: {Времето за чистене }”**
2. Има ли изненада или не:

* Ако братята **СА изненадали бащата (time left > 0)**: “Yes, **there is a surprise -** time left - {остатък} hours.” – резултата трябва да е **закръглен към по-малко цяло число** (**пр. 1.90 -> 1**).
* Ако братята **НЕ СА изненадали бащата**: “No, **there isn’t a surprise** - shortage of time - {недостиг} hours.” – резултата трябва да е **закръглен към по-голямо цяло число** (**пр. 1.10 -> 2**).

### Примерен вход и изход

|  |  |  |
| --- | --- | --- |
| **Вход** | **Изход** | **Обяснения** |
| 3  4  5  3.6 | Cleaning time: 1.47  Yes, there is a surprise - time left -> 2 hours. | **общо време** =1 / (1/3+1/4+1/5)= **1.27659574468085 часа**  **време с почивка** = 1.27659574468085 \*0.15= **1.46808510638298 часа**  3.6 - 1.46808510638298 = 2.13191489361702 **= 2 часа остават** |
| **Вход** | **Изход** | **Обяснения** |
| 3.2  2.5  1.24  0.5 | Cleaning time: 0.76  No, there isn't a surprise - shortage of time -> 1 hours. |  |

## \* Стипендии

*Втора задача от изпитa на 03 септември 2017. Тествайте решението си* [***тук***](https://judge.softuni.bg/Contests/Practice/Index/750#1)*.*

Учениците могат да кандидатстват за **социална стипендия** или за **стипендия за отличен успех**. Изискване за социална стипендия – **доход на член от семейството по-малък от минималната работна заплата и успех над 4.5**. Размер на социалната стипендия – **35% от минималната работна заплата**. Изискване за стипендия за отличен успех – **успех над 5.5, включително**. Размер на стипендията за отличен успех – **успехът на ученика, умножен по коефициент 25**.

Напишете програма, която при въведени **доход**, **успех** и **минимална** **работна** **заплата**, дава информация дали ученик има право да получава стипендия, и стойността на стипендията, която **е по-висока** за него.

### Вход

Потребителят въвежда **3 числа**, по едно на ред:

1. **Доход в лева** – **реално число в интервала [0.00..6000.00];**
2. Среден успех– **реално число в интервала [2.00...6.00];**
3. Минимална работна заплата – реално **число в интервала [0.00..1000.00];**

### Изход

* Ако ученикът **няма право да получава стипендия**, се извежда:

**"You cannot get a scholarship!"**

* Ако ученикът **има право да получава социална стипендия** и тя **е по-висока** от стипендията за отличен успех:  
  **"You get a Social scholarship {стойност на стипендия} BGN"**
* Ако ученикът **има право да получава стипендия за отличен успех** и тя е **по-висока** или **равна** по стойност на социалната стипендия за него:  
  **"You get a scholarship for excellent results {стойност на стипендията} BGN"**

**Резултатът се закръгля до по-малкото цяло число.**

### Примерен вход и изход

|  |  |  |
| --- | --- | --- |
| **Вход** | **Изход** | **Обяснения** |
| 480.00  4.60  450.00 | **You cannot get a scholarship!** | Доходът от 480 лв. е по-висок от минималната работна заплата 450 лв. → ученикът не може да получава социална стипендия.  Успех 4.60 < 5.50 → ученикът не може да получава стипендия за отличен успех. |
| 300.00  5.65  420.00 | **You get a Social scholarship 147 BGN** | 300 лв. < 420 лв. и 5.65 >4.50 → ученикът може да получава социална стипендия 35% \* 420 лв. = 147 лв.  Успех 5.65 > 5.50 → ученикът може да получава стипендия за отличен успех 5.65 \* 25 = 141.25 лв.  147 лв. > 141.25 лв. → ученикът ще получава социална стипендия. |

## \* Хореография

*Втора задача от изпитa на 23 юли 2017. Тествайте решението си* [***тук***](https://judge.softuni.bg/Contests/Practice/Index/715#1)*.*

Група танцьори се подготвя за финално състезание. Те трябва да научат нова хореография. Танца се състои от **N - на брой стъпки**, които се **разпределят между танцьорите**. Цялата хореография трябва да се научи за **определен брой дни**. **Всички** танцьори могат да научат **не повече от 13% от общите стъпки на ден**. Да се напише програма която пресмята **дали танцьорите ще успеят да научат новия танц** и **по колко процента от стъпките следва да научи всеки един от тях**.

**При изчисляване на процента стъпки на ден, числото трябва да се закръгли към най – близкото цяло число нагоре.**

### Вход

От конзолата се четат **3 реда**:

1. **Брой стъпки – цяло число в интервала [1 … 100 000];**
2. **Брой танцьори – цяло число в интервала [1 … 50];**
3. **Брой дни за учене – цяло число в интервала [1 … 31];**

### Изход

Отпечатването на конзолата зависи от резултата:

* Ако общия процент стъпки **са по-малко или равни на 13%** отпечатваме:
  + **"Yes, they will succeed in that goal! {процент стъпки които трябва да научи всеки един танцьор на ден}%."**
* Ако общия процент стъпки **са повече от 13%** отпечатваме:
  + **"No, They will not succeed in that goal! Required {процент стъпки, които трябва да научи всеки един танцьор на ден}% steps to be learned per day."**

И двата отговора трябва да са форматирани **до втория** знак след десетичната запетая.

### Примерен вход и изход

|  |  |  |
| --- | --- | --- |
| **вход** | **изход** | **коментари** |
| 10464  20  20 | Yes, they will succeed in that goal! 0.25%. | Стъпки на ден: (10464 / 20) / 10464 = 5%  Процент стъпки за всеки танцьор: 5 / 20 = 0.25% |
| 55555  30  7 | No, They will not succeed in that goal! Required 0.50% steps to be learned per day. | Стъпки на ден: (55555 / 7) / 55555 = 14.28..% = 15%  Процент стъпки за всеки танцьор: 15 / 30 = 0.50% |

## \* Световен рекорд по плуване

*Втора задача от изпитa на 25 юни 2017. Тествайте решението си* [***тук***](https://judge.softuni.bg/Contests/Practice/Index/642#1)*.*

Иванчо решава да подобри Световния рекорд по плуване на дълги разстояния. **На конзолата се въвежда рекордът в секунди, който Иванчо трябва да подобри, разстоянието в метри, което трябва да преплува и времето в секунди, за което плува разстояние от 1 м.** Да се напише програма, която изчислява дали се е справил със задачата, като се има предвид, че: **съпротивлението на водата го забавя на всеки 15 м. с 12.5 секунди. Да се изчисли времето в секунди, за което Иванчо ще преплува разстоянието и разликата спрямо Световния рекорд.**

Когато се изчислява колко пъти Иванчо ще се забави в резултат на съпротивлението на водата **резултатът трябва да се закръгли надолу до най-близкото цяло число.**

### Вход

От конзолата се четат **3 реда**:

1. **Рекордът в секунди – реално число в интервала [0.00 … 100000.00];**
2. **Разстоянието в метри – реално число в интервала [0.00 … 100000.00];**
3. **Времето в секунди, за което плува разстояние от 1 м.** **– реално число в интервала [0.00 … 1000.00];**

### Изход

Отпечатването на конзолата зависи от резултата:

* Ако **Иванчо е подобрил Световния рекорд** отпечатваме:
  + **"** **Yes, he succeeded! The new world record is {времето на Иванчо} seconds."**
* Ако **НЕ е подобрил рекорда** отпечатваме:
  + **"No, he failed! He was {недостигащите секунди} seconds slower."**

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

### Примерен вход и изход

|  |  |  |
| --- | --- | --- |
| **Вход** | **Изход** | **Обяснения** |
| 10464  1500  20 | No, he failed! He was 20786.00 seconds slower. | Иванчо трябва да преплува **1500 м**.: 1500 \* 20 = **30000 сек.**  **На всеки** **15 м. към времето му се добавят 12.5 сек**.:  (1500 / 15) \* 12.5 = **1250 сек.**  **Общо време:** 30000 + 1250 = **31250 сек.**  10464 < 31250  **Времето, което не му е стигнало за да подобри рекорда:**  31250 – 10464 = **20786 сек.** |
| **Вход** | **Изход** | **Обяснения** |
| 55555.67  3017  5.03 | Yes, he succeeded!The new world record is 17688.01 seconds. | Иванчо трябва да преплува **3017 м**.: 3017 \* 5.03 = **15175.51 сек.**  **На всеки** **15 м. към времето му се добавят 12.5 сек**.:  3017/ 15 = 201 \* 12.5 = **2512.50 сек.**  **Общо време:** 15175.51 + 2512.50 = **17688.01 сек.**  **Рекордът е подобрен:** 55555.67 > 17688.01 |

## \* Магазин за детски играчки

*Втора задача от изпитa на 07 май 2017. Тествайте решението си* [***тук***](https://judge.softuni.bg/Contests/Practice/Index/540#1)*.*

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** |