# Лаб: Повторения с цикли – while-цикъл

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

**Тествайте** решенията си в **judge системата**: <https://judge.softuni.bg/Contests/Compete/Index/1014>

## Число в диапазона [1…100]

Напишете програма, която въвежда цяло положително **число** n **в диапазона [1…100]**. При въвеждане на число извън посочения диапазон, да се отпечата съобщение за грешка и потребителят да се подкани **да въведе ново число**.

### Примерен вход и изход

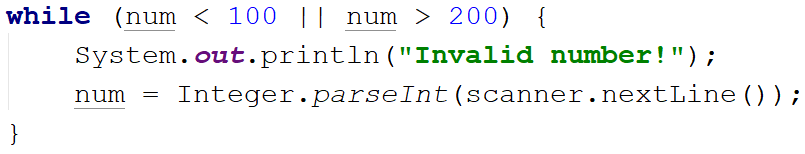
|  |
| --- |
| **Вход / Изход** |
| **35**  The number is: 35 |
| **105**  Invalid number!  **0**  Invalid number!  **-200**  Invalid number!  **77**  The number is: 77 |

### Насоки

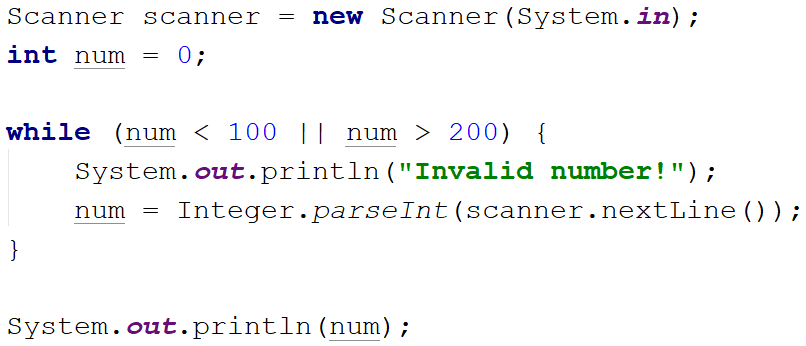
1. Инициализирайте променлива num, в която да пазите стойността на текущото число:



1. В while цикъл, до получаване на валидно число, четете ново:



1. Когато намерите **валидно число** го **принтирайте**:



## Редица числа 2k+1

Напишете програма, която чете число n, въведено от потребителя,и отпечатва **всички числа ≤** n **от редицата**: **1, 3, 7, 15, 31**, …. Всяко следващо число се изчислява като умножим **предишното** с **2** и добавим **1**.

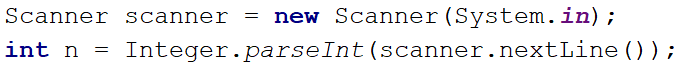
### Примерен вход и изход

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 3 | 1  3 | 8 | 1  3  7 | 17 | 1  3  7  15 | 31 | 1  3  7  15  31 |

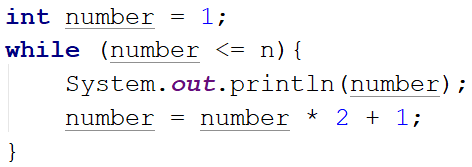
**Тествайте** решението си в **judge системата**: <https://judge.softuni.bg/Contests/Compete/Index/1014#7>

### Насоки

1. Прочетете n – цяло число, което представлява краят на редицата



1. В while цикъл до достигане на n, пресмятайте ново число от редицата и го принтирайте:



## Баланс по сметка

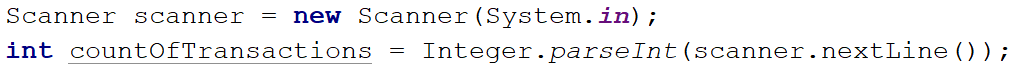
Напишете програма, която пресмята **колко общо пари** има в сметката, след като направите **определен брой вноски**. На първия ред ще получите **колко вноски трябва да се направят**. На всеки следващ ред ще получавате сумата, която трябва да внесете в сметката, **докато не се достигне броя вноски**. При всяка получена сума на конзолата трябва да се извежда **"Increase: "** + сумата и тя да се **прибавя в сметката**. Ако получите число **по-малко от 0** на конзолата трябва да се изведе **"Invalid operation!"** и **програмата да приключи**. Когато програмата приключи трябва да се принтира **"Total: "** + общата сума в сметката закръглена до втория знак след десетичната запетая.

### Примерен вход и изход

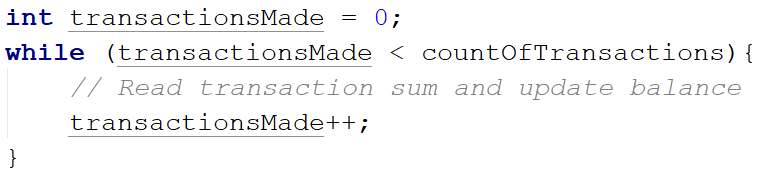
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |
| 3  5.51  69.42  100 | Increase: 5.51  Increase: 69.42  Increase: 100  Total: 174.93 | 5  120  45.55  -150 | Increase: 120  Increase: 45.55  Invalid operation!  Total: 165.55 |

### Насоки

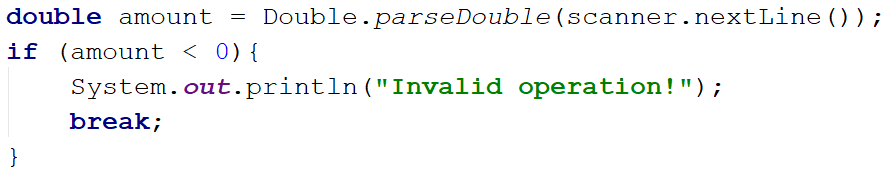
1. Прочетете **броя на вноските** (**транзакциите**), които ще бъдат направени:



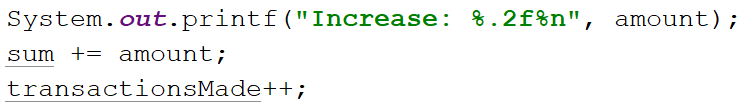
1. В while цикъл, докато броя на **направените транзакции** е по-малък от тези, които трябва да се направят, четете сума на текущата транзакция и извършвайте пресмятания и проверки с нея:



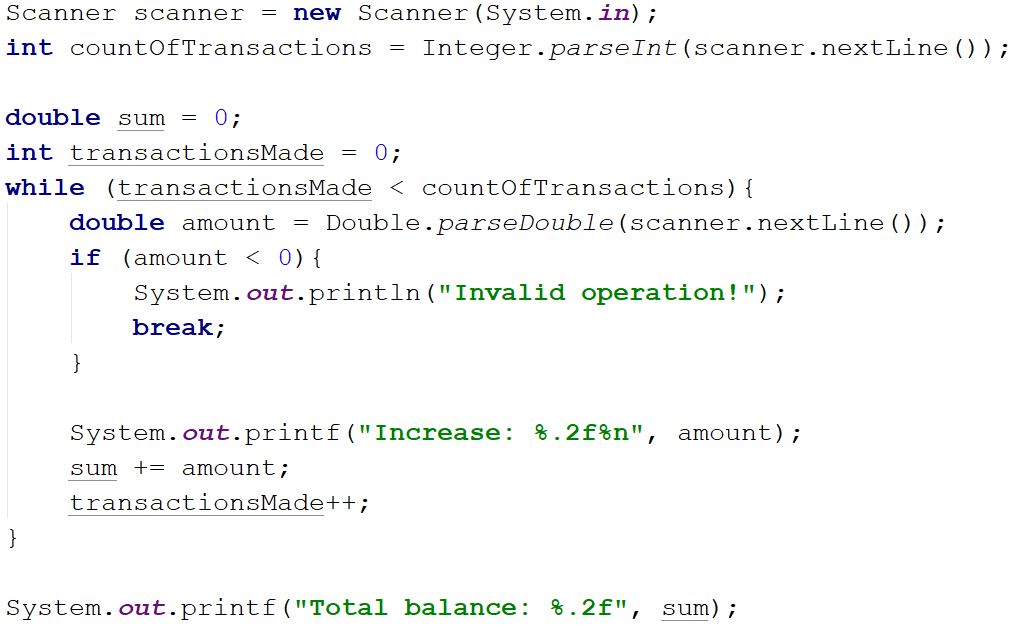
1. Четете сумата на текущата транзакция. Валидирайте я и принтирайте подходящо съобщение, ако тя е по-малка от 0:



1. Ако сумата на транзакцията е положителна, принтирайте съобщение за промяна в баланса на сметката и увеличете нейната стойност. Увеличете броя на напаравените транзакции:



1. Когато достигнете броя на нужните транзакции, принтирайте **крайния баланс на сметката**:



## Редица цели числа

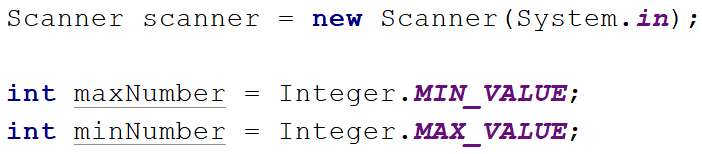
Напишете програма, която чете **цели числа**, **докато** не се получи командата **"**END**"**. Принтирайте **най-голямото** и **най-малкото** число сред въведените.

### Примерен вход и изход

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |
| 10  20  304  0  50  END | Max number: 304  Min number: 0 | 250  5  2  0  100  1000  END | Max number: 1000  Min number: 0 |

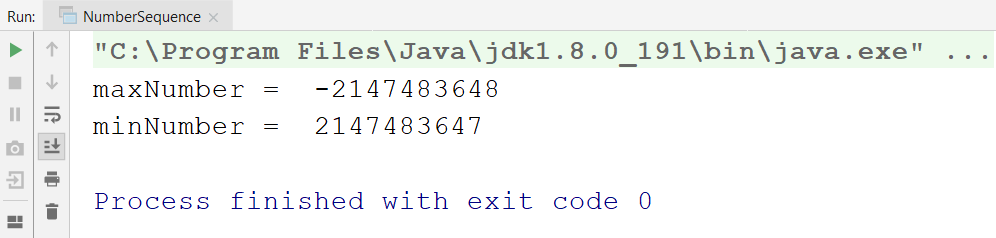
### Насоки

1. Създайте обект Scanner, с който да четете от конзолата. Инициализирайте 2 променливи:
   1. maxNumber - присвоете стойност най-малката възможна за типа int чрез Integer.MIN\_VALUE, за да сте сигурни, че **въведеното число винаги ще е по-голямо от най-малкото възможно за типа данни**
   2. minNumber - присвоете стойност най-голямата възможна за типа int чрез Integer.MAX\_VALUЕ, за да сте сигурни, че **въведеното число винаги ще е по-малко от най-голямото за типа данни**

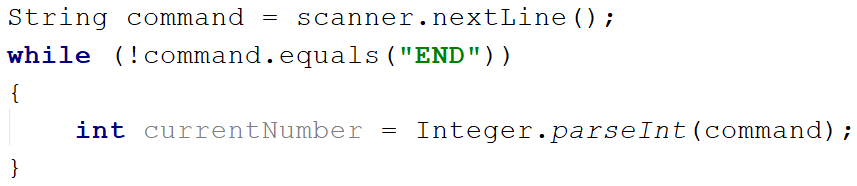


Повече информация за **стойностните ограничения** на типовете данни можете да прочетете [тук](https://docs.oracle.com/javase/tutorial/java/nutsandbolts/datatypes.html).

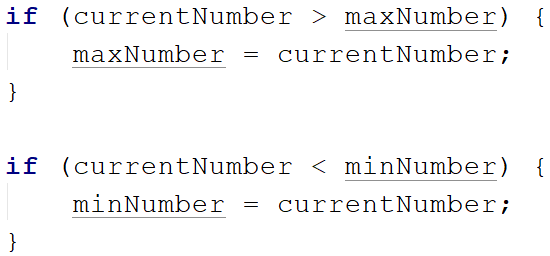
Принтирайте двете променливи, за да проверите запаметените стойности в тях:



1. Прочетете първите входни данни от конзолата и ги запаметете в променлива от тип string - command. В while цикъл до получаване на стойност "END" преобразувайте прочетената стойност в число:

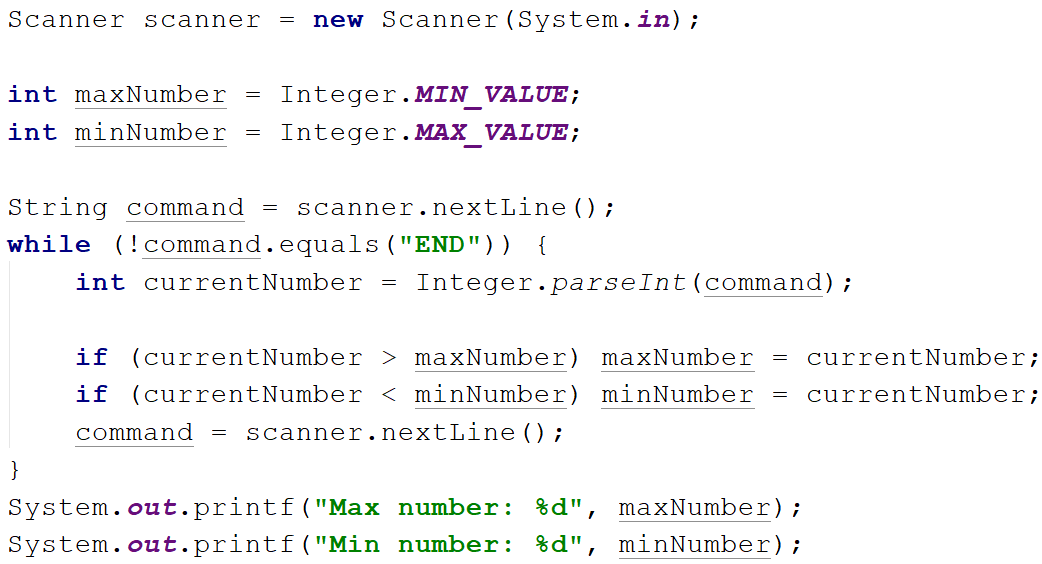


1. Сравнете прочетеното число с най-голямото и най-малкото до момента:
   1. Ако е по-голямо от най-голямото – maxNumber, приравнете стойността на maxNumber с тази на прочетеното число
   2. Ако е по-малко от най-малкото – minNumber, приравнете стойността на minNumber с тази на прочетеното число



Накрая прочетете нова стойност за променливата command.

1. След получаване на стойност END, принтирайте стойностите на най-малкото и най-голямото число:



## Стъпки

Габи иска да започне здравословен начин на живот и си е поставила за цел да върви **10 000 стъпки всеки ден**. Някои дни обаче е много уморена от работа и ще иска да се прибере преди да постигне целта си. Напишете програма, която **чете от конзолата по колко стъпки изминава** тя всеки път като излиза през деня и **когато постигне целта си да се изписва** "**Goal reached! Good job!**"

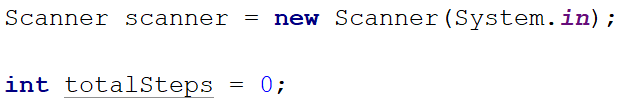
Ако иска да се **прибере преди това**, тя ще въведе **командата** "Going home" и **ще въведе стъпките, които е извървяла докато се прибира**. След което, ако не е успяла да постигне целта си, на конзолата трябва да се изпише: "{разликата между стъпките} more steps to reach goal."

### Примерен вход и изход

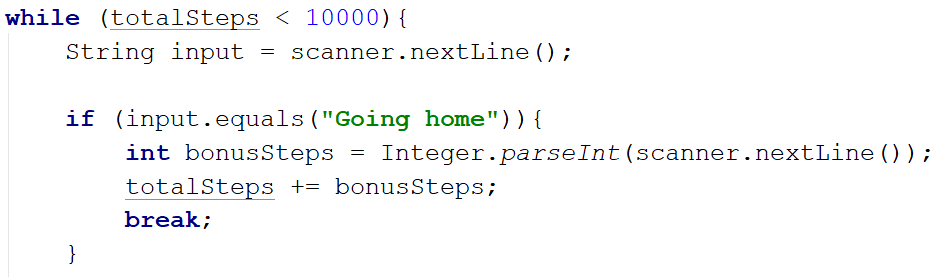
|  |  |  |  |
| --- | --- | --- | --- |
| **Вход** | **Изход** | **Вход** | **Изход** |
| 1000  1500  2000  6500 | Goal reached! Good job! | 1500  300  2500  3000  Going home  200 | 2500 more steps to reach goal. |
| **Вход** | **Изход** | **Вход** | **Изход** |
| 1500  3000  250  1548  2000  Going home  2000 | Goal reached! Good job! | 125  250  4000  30  2678  4682 | Goal reached! Good job! |

### Насоки

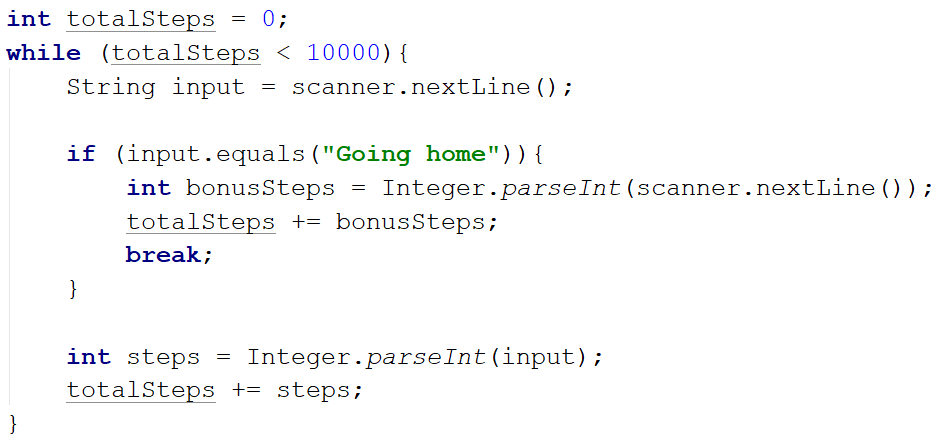
1. Създайте обект Scanner, с който да четете от конзолата, и променлива totalSteps, в която да добавяте крачките, които Габи е изминала:



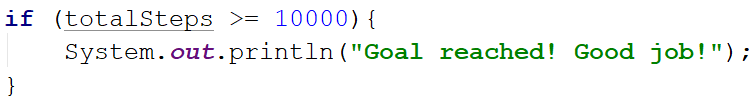
1. В while цикъл, докато Габи не е достигнала **10 000** стъпки, четете вход въведен от потребителя.
   1. Ако въведения вход е "Going home", прочетете стъпките, които Габи изминава преди да се прибере и ги добавете към общия брой стъпки и прекъснете цикъла:



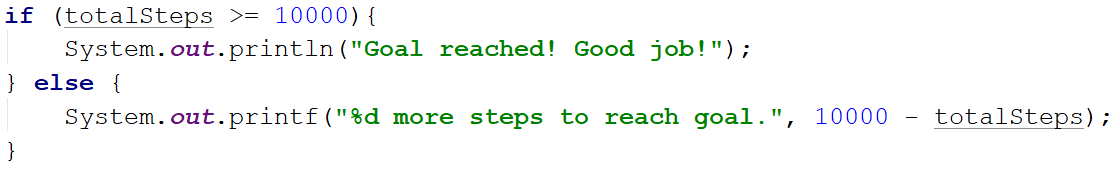
* 1. Ако въведената стойност от потребителя не е "Going home", добавете полученото към общия брой стъпки и продължете към следващата итерация на цикъла:



1. След като Габи е достигнала крачките или се е прибрала вкъщи, принтирайте изхода според това дали е достигнала целта си или не:
   1. Ако **е достигнала** 10 000 крачки, принтирайте:



* 1. Ако **не е достигнала** целта си:



## Завършване

Напишете програма, която изчислява **средната оценка** на ученик от цялото му обучение. На първия ред ще получите **името на ученика**, а на всеки следващ ред неговите годишни оценки. Ученикът преминава в следващия клас, **ако годишната му оценка е** **по-голяма или равна на 4.00**. **Ако оценката му е под 4.00**, той ще повтори класа.

При успешно завършване на **12-ти** клас да се отпечата:

"{**име на ученика**} graduated. Average grade: {**средната оценка от цялото обучение**}"

**Стойността трябва да бъде форматирана до втория знак** след десетичната запетая.

### Примерен вход и изход

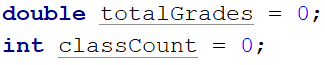
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **вход** | **изход** | **вход** | **изход** | **вход** | **изход** |
| Pesho  4  5.5  6  5.43  4.5  6  5.55  5  6  6  5.43  5 | Pesho graduated. Average grade: 5.37 | Pesho  5  5  5  6  5.5  5  6  5.44  5  5  5  5  6  5.45 | Pesho graduated. Average grade: 5.25 | Ani  5  5.32  6  5.43  5  6  5.5  4.55  5  6  5.56  6 | Ani graduated. Average grade: 5.45 |

### Насоки

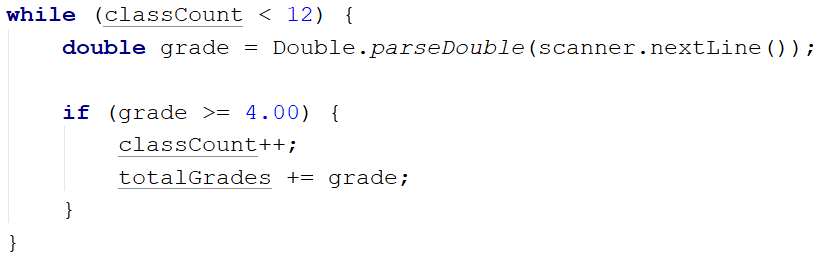
1. Прочетете името на ученика – текст въведен от конзолата:



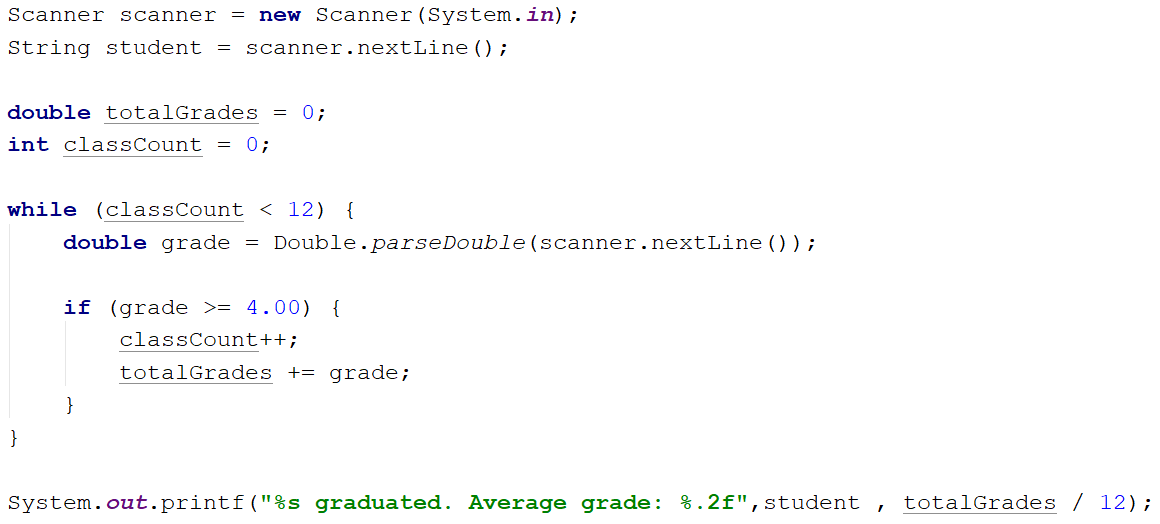
1. Направете променливи totalGrades и classCount, в които да пазите сумата на всички оценки и броя на класовете, които е преминал ученика:



1. Направете while цикъл, в който докато ученикът не стигне 12 клас, четете оценка. Запазете стойността й в променлива grade. Проверете дали стойността е **по-голяма** или **равна** **на** **4** и ако е, увеличете стойността на classCount и добавете оценката към totalGrades**:**



1. Принтирайте изхода според условието на задачата, като изчислите и средната оценка:



## Завършване - част 2

Напишете програма, която изчислява **средната оценка** на ученик от цялото му обучение. На първия ред ще получите **името на ученика**, а на всеки следващ ред неговите годишни оценки. Ученикът преминава в следващия клас, **ако годишната му оценка е по-голяма или равна на 4.00**. **Ако ученикът бъде скъсан повече от един път, то той бива изключен и програмата приключва,** като се отпечатва **името на ученика и в кой клас бива изключен**.

При успешно завършване на **12-ти** клас да се отпечата :

"{**име на ученика**} graduated. Average grade: {**средната оценка от цялото обучение**}"

**В случай, че ученикът е изключен от училище, да се отпечата:**

"{**име на ученика**} has been excluded at {**класа, в който е бил изключен**} grade"

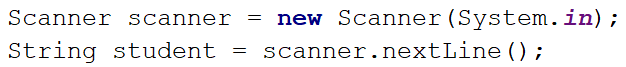
**Стойността трябва да бъде форматирана до втория знак** след десетичната запетая.

### Примерен вход и изход

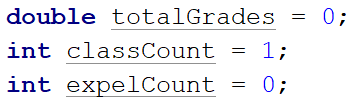
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |
| Gosho  5  5.5  6  5.43  5.5  6  5.55  5  6  6  5.43  5 | Gosho graduated. Average grade: 5.53 | Mimi  5  6  5  6  5  6  6  2  3 | Mimi has been excluded at 8 grade |

### Насоки

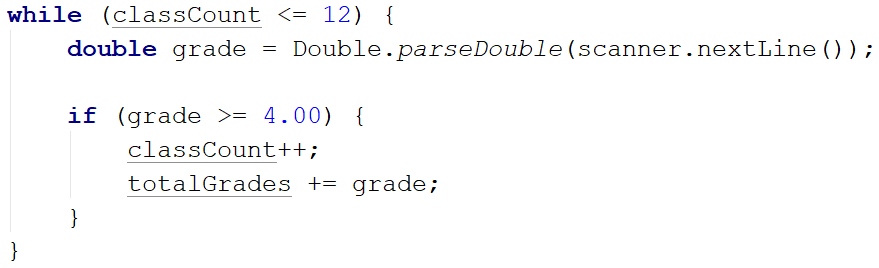
1. Прочетете името на ученика – текст, въведен от конзолата:



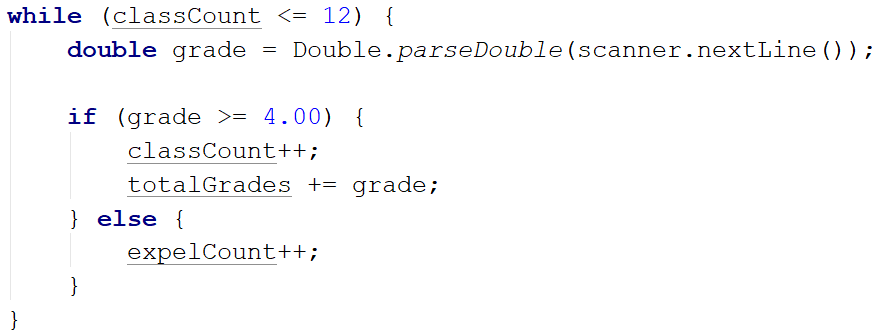
1. Инициализирайте **3** променливи – totalGrades, classCount, еxpelCount, в които да пазите сумата на всички оценки, броя на преминатите класове и броя на пътите, в които ученикът е скъсан:



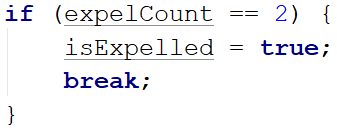
1. В while цикъл докато ученикът не достигне 12 клас, четете оценка и сравнете дали е **по-голяма** или **равна на 4**. Ако проверката е вярна, увеличете броя на преминатите класове и сумирайте оценката:



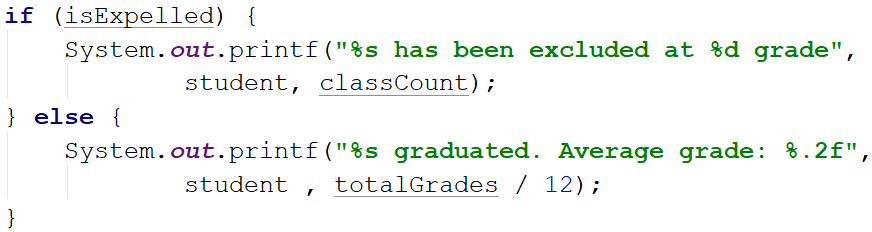
1. Ако оценката **не е по-голяма от 4**, увеличете броя на пътите, в които ученикът е скъсан:



1. Проверете дали пътите, в които ученикът е скъсан, не са станали 2. Ако е вярно, прекъснете цикъла и запаметете резултата в променлива isExpelled:



1. Принтирайте резултат според това дали ученикът е скъсан или не:



# Примерна изпитна задача

## \*Преместване

На осемнадесетия си рожден ден на Хосе взел решение, че ще се изнесе да живее на квартира. Опаковал багажа си в **кашони** и намерил подходяща обява за апартамент под наем. Той започва да пренася своя багаж **на части**, защото не може да пренесе целия наведнъж. Има ограничено **свободно пространство** в новото си жилище, където може да разположи вещите, така че мястото да бъде подходящо за живеене.

Напишете **програма, която изчислява свободния обем от жилището на Хосе, който остава след като пренесе багажа си.**

**Бележка: Един кашон е с точни размери: 1m. x 1m. x 1m.**

**Вход**

Потребителят въвежда следните данни на отделни редове:

1. **Широчина на свободното пространство - цяло число в интервала [1...1000]**
2. **Дължина на свободното пространство - цяло число в интервала [1...1000]**
3. **Височина на свободното пространство - цяло число в интервала [1...1000]**
4. На следващите редове **(до получаване на команда "Done") -** **брой кашони, които се пренасят в квартирата - цели числа в интервала [1...10000]**

**Програмата трябва да приключи прочитането на данни при команда "Done"** **или ако свободното място свърши.**

**Изход**

Да се **отпечата** на конзолата **един** от следните редове:

* Ако стигнете до командата **"Done"** и има още свободно място:

**"{брой свободни куб. метри} Cubic meters left."**

* Ако свободното място свърши преди да е дошла команда **"**Done**":**

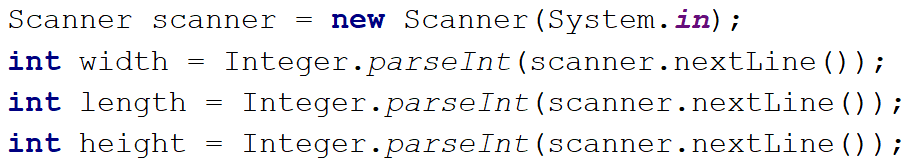
**"No more free space! You need {брой недостигащи куб. метри} Cubic meters more."**

**Примерен вход и изход**

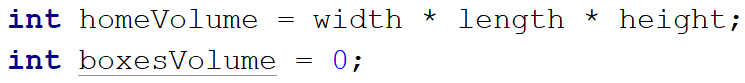
|  |  |  |
| --- | --- | --- |
| **Вход** | **Изход** | **Обяснение** |
| 10  10  2  20  20  20  20  122 | No more free space! You need 2 Cubic meters more. | 10 \* 10 \* 2 = **200 кубични метра.** 20 + 20 + 20 + 20 + 122 = **202** **кубични метра.** 200 - 202 = **2 недостигащи** кубични метра |
| 10  1  2  4  6  **Done** | 10 Cubic meters left. | 10 \* 1 \* 2 = **20 кубични метра.** 4 + 6 = **10** **кубични метра.** 20 - 10 = **10 кубични метра.** |

### Насоки

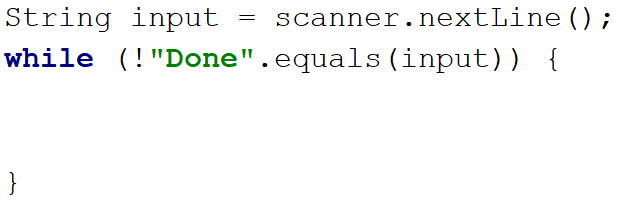
1. Прочетете входните данни от конзолата – размерите на жилището



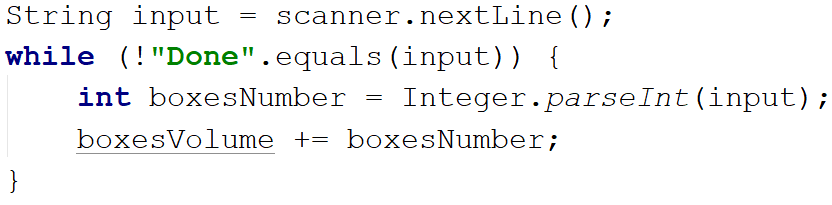
1. Изчислете обема на жилището и направете променлива boxesVolume, в която да пазите обема на всички кутии:



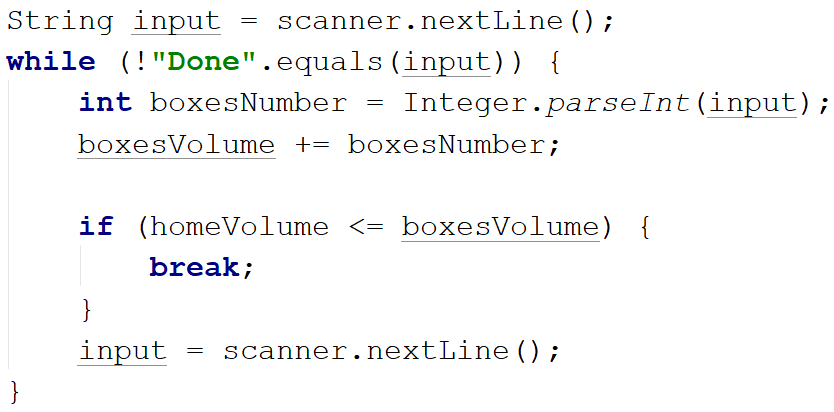
1. Прочетете първите входни данни от потребителя и направете while цикъл, който да прави повторения до получаване на текста “Done!”:



1. Превръщайте прочетената стойност в число – броя на кашоните, които Хосе пренася. Тъй като един кашон е с размерите 1м x 1м х 1м, добавяйте полученото число към променливата за общия обем на кашоните boxesVolume:



1. Ако обема на кутиите достигне размер е **по-голям или равен на обема на жилището** на Хосе, прекъснете цикъла. В противен случай прочетете отново входните данни и продължете към следващата итерация на цикъла:



1. След като прочетете всички входни данни и направите нужните изчисления, проверете **дали обема на кутиите е по-голям от обема на жилището** на Хосе и принтирайте подходящия изход според условието на задачата:

