# Лаб: Повторения с цикли – for-цикъл

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/1015>

## Числа от 1 до 100

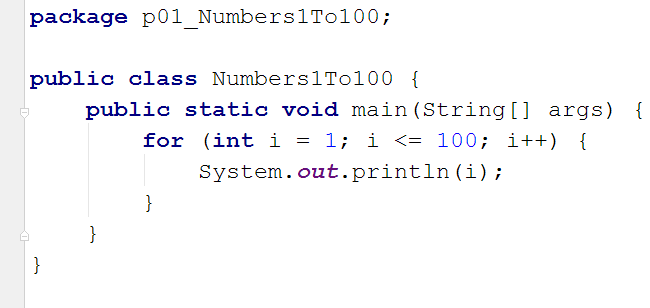
Напишете програма, която отпечатва числата от 1 до 100, по едно на ред.

### Примерен вход и изход

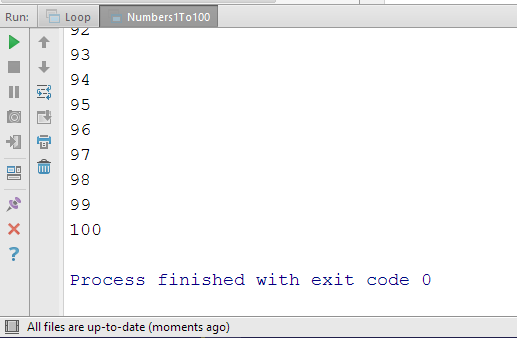
|  |  |
| --- | --- |
| **вход** | **изход** |
| (няма) | 1  2  3  …  98  99  100 |

### Насоки

1. Създайте for цикъл с начална стойност на контролната променлива i = 1. Направете крайната стойност по-малка или равна на **100**. В блока от код на цикъла принтирайте i:



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



## Всички латински букви

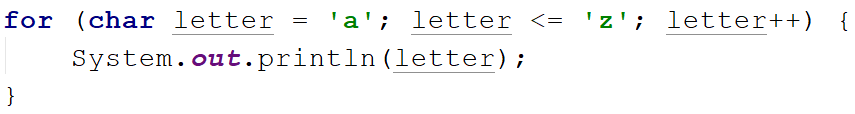
Напишете програма, която отпечатва всички букви от латинската азбука: a, b, c, …, z.

**Тествайте** решението си в **judge системата**:

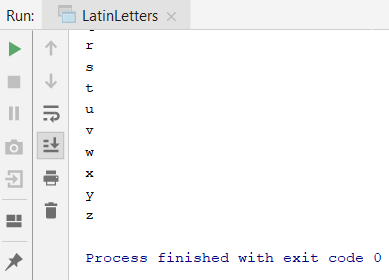
**Подсказка**: можете да завъртите for-цикъл от 'a' до 'z' (освен числа може да въртите в цикъл и букви).

### Насоки

1. Създайте for цикъл с тип на контролната променлива char и начална стойност 'a' и крайна 'z'. На всяка итерация увеличавайте стойността с 1. В блока от код на цикъла принтирайте стойността на контролната променлива letter.



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



## Сумиране на числа

Да се напише програма, която **чете** n-на брой **цели числа**, въведени от потребителя **и ги сумира**.

* От първия ред на входа се въвежда броят числа n.
* От следващите n реда се въвежда по едно цяло число.

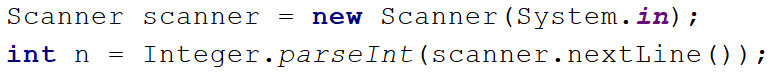
Програмата трябва да прочете числата, да ги сумира и да отпечата сумата им.

### Примерен вход и изход

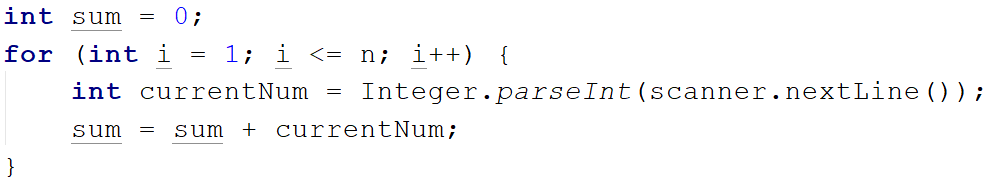
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 2  10  20 | 30 | 3  -10  -20  -30 | -60 | 4  45  -20  7  11 | 43 | 1  999 | 999 | 0 | 0 |

### Насоки

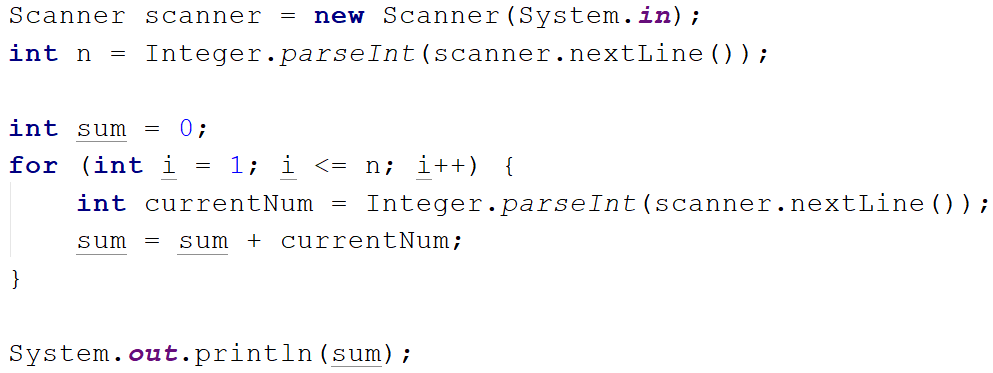
1. Прочетете едно цяло число n (броят числа, които предстои да бъдат въведени):



1. Във for цикъл четете всеки път ново цяло число, докато сте прочели n на брой числа. Сумирайте всяко въведено число "currentNum" в променлива sum, в която ще пазите сумата от всички въведени числа:



1. Накрая в sum трябва да се е запазила сумата на прочетените числа. Отпечатайте я:



## Най-голямо число

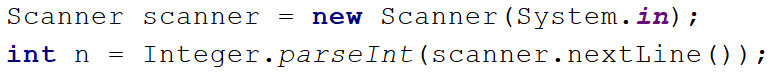
Напишете програма, която чете n-на брой **цели числа** (n > 0), въведени от потребителя, и намира **най-голямото** измежду тях. Първо се въвежда броят числа n, а след това самите n числа, по едно на ред.

### Примерен вход и изход

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 2  100  99 | 100 | 3  -10  20  -30 | 20 | 4  45  -20  7  99 | 99 | 1  999 | 999 | 2  -1  -2 | -1 |

### Насоки

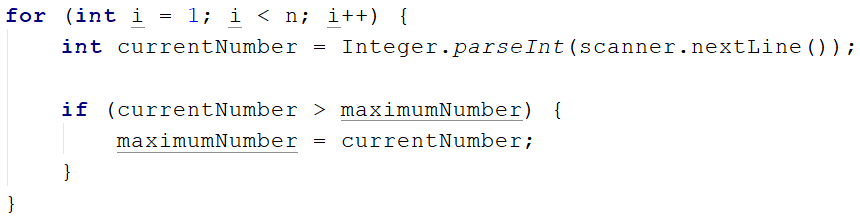
1. Първо прочетете едно число цяло n (броят числа, които предстои да бъдат въведени):



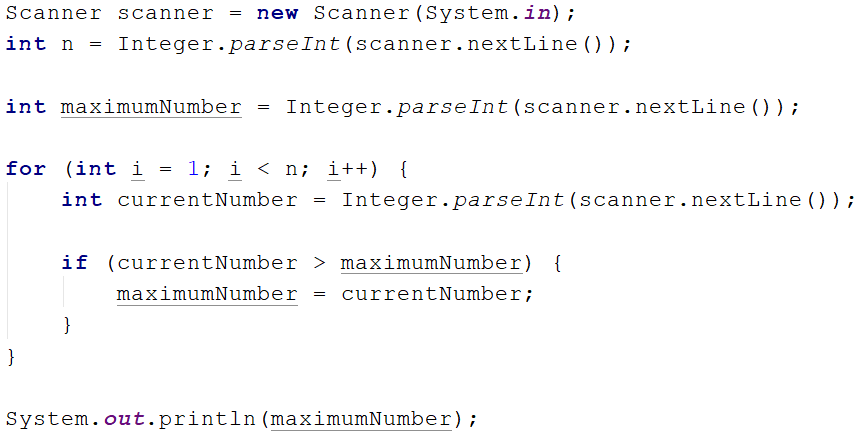
1. Въведете от конзолата първото число. Сложете текущият максимум maximumNumber да е прочетеното число:



1. Във for цикъл от 1 до n пъти прочетете по едно цяло число currentNumber. Ако прочетеното число currentNumber е по-голямо от текущият максимум maximumNumber, запомнете currentNumber в maximumNumber:

`

1. Накрая в maximumNumber трябва да се е запазило най-голямото число. Отпечатайте го:



## Най-малко число

Напишете програма, която чете n-на брой **цели числа** (n > 0), въведени от потребителя и намира **най-малкото** измежду тях. Първо се въвежда броят числа n, а след това самите n числа, по едно на ред.

### Примерен вход и изход

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 2  100  99 | 99 | 3  -10  20  -30 | -30 | 4  45  -20  7  99 | -20 | 1  999 | 999 | 2  -1  -2 | -2 |

## Лява и дясна сума

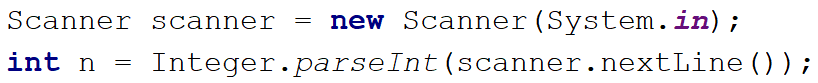
Да се напише програма, която чете **2 \*** n-на брой цели числа, подадени от потребителя, и проверява дали **сумата на първите** n **числа** (лява сума) е равна на **сумата на вторите** n **числа** (дясна сума). При равенство печата "Yes, sum = " + **сумата**; иначе печата "No, diff = " + **разликата**. Разликата се изчислява като положително число (по абсолютна стойност).

### Примерен вход и изход

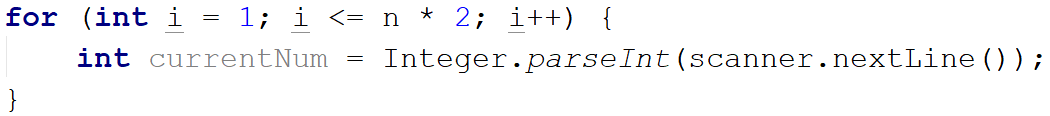
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** | **коментар** |  | **вход** | **изход** | **коментар** |
| 2  10  90  60  40 | Yes, sum = 100 | 10+90 = 60+40 = 100 | 2  90  9  50  50 | No, diff = 1 | 90+9 ≠ 50+50  Difference = |99-100| = 1 |

### Насоки

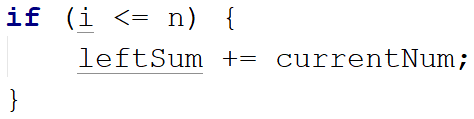
1. Прочетете цяло число n:



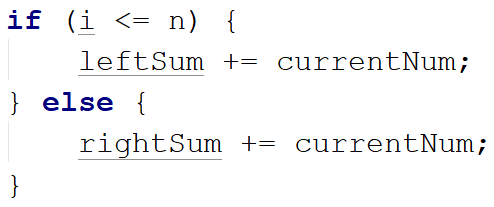
1. Във for цикъл от 1 до n \* 2 пъти четете по едно цяло число:



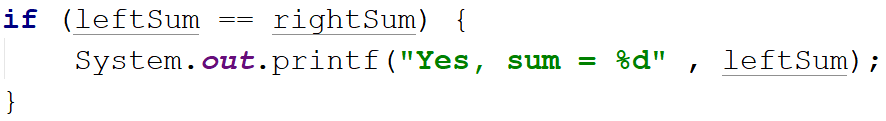
1. Според стойността на контролната променлива сумирайте в подходящата сума:
   * Ако числото е в лявата половина (i <= n) го сумирайте в променлива leftSum:



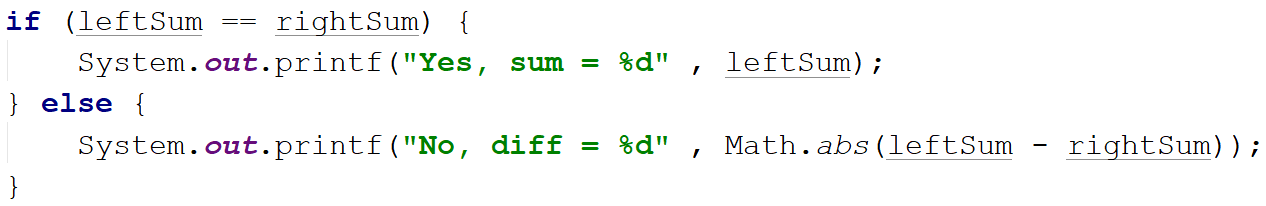
* + Ако числото е в дясната половина (n < i <= n \* 2) го сумирайте в променлива rightSum:



1. След като сте прочели всички числа, изчислете **разликата** между сумите:
   * Ако сумите са равни, принтирайте:



* + Ако сумите са различни, принтирайте **абсолютната стойност** на разликата между тях:



## Четна / нечетна сума

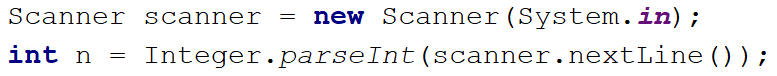
Да се напише програма, която чете n на брой цели числа, подадени от потребителя, и проверява дали **сумата от числата на четни позиции** е равна на **сумата на числата на нечетни позиции**. При равенство да се отпечата "Yes" + **сумата**; иначе да се отпечата "No" + **разликата**. Разликата се изчислява по абсолютна стойност.

### Примерен вход и изход

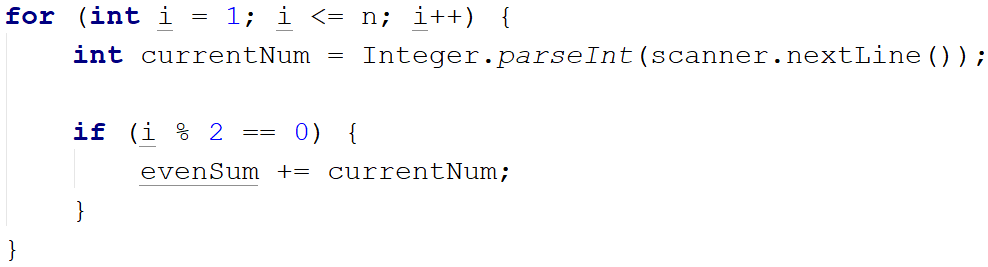
|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** | **коментар** |  | **вход** | **изход** | **коментар** |  | **вход** | **изход** | **коментар** |
| 4  10  50  60  20 | Yes  Sum = 70 | 10+60 = 50+20 = 70 | 4  3  5  1  -2 | No  Diff = 1 | 3+1 ≠ 5-2  Diff = |4-3| = 1 | 3  5  8  1 | No  Diff = 2 | 5+1 ≠ 8  Diff = |6-8| = 2 |

### Насоки

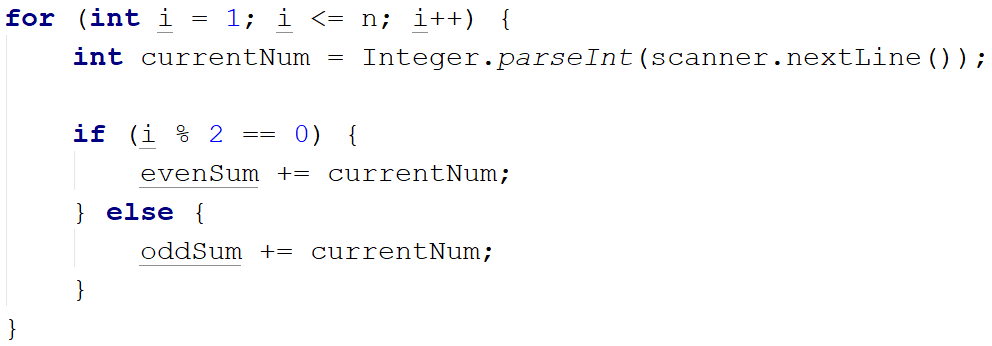
1. Прочетете едно **цяло** **число** от конзолата - броя на числата, които трябва да бъдат въведени:



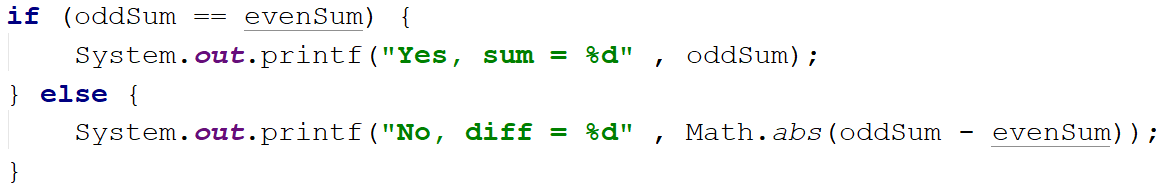
1. Във for цикъл въведете числата едно по едно. Създайте променливи oddSum и evenSum, в които ще пазите сумите на числата на четна и нечетна позиция
   1. Ако контролната променлива на цикъла е с **четна** стойност, сумирайте прочетеното число в **сумата на четните** числа:



* 1. Ако контролната променлива на цикъла е с нечетна стойност, сумирайте прочетеното число в **сумата на нечетните** числа:



1. Както в предходната задача, изчислете абсолютна стойност на разликата и отпечатайте резултата ("Yes" + **сумата** при разлика 0 или "No" + **разликата** в противен случай)



## Сумиране на гласните букви

Да се напише програма, която чете **текст** (string), въведен от потребителя, изчислява и отпечатва **сумата от стойностите на гласните букви** според таблицата по-долу:

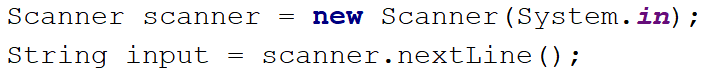
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| буква | a | e | i | o | u |
| стойност | 1 | 2 | 3 | 4 | 5 |

### Примерен вход и изход

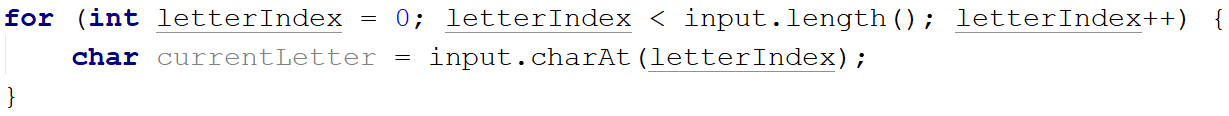
|  |  |  |
| --- | --- | --- |
| **вход** | **изход** | **коментар** |
| hello | 6 | e + o = 2 + 4 = 6 |
| hi | 3 | i = 3 |
| bamboo | 9 | a + o + o = 1 + 4 + 4 = 9 |
| beer | 4 | e + e = 2 + 2 = 4 |

### Насоки

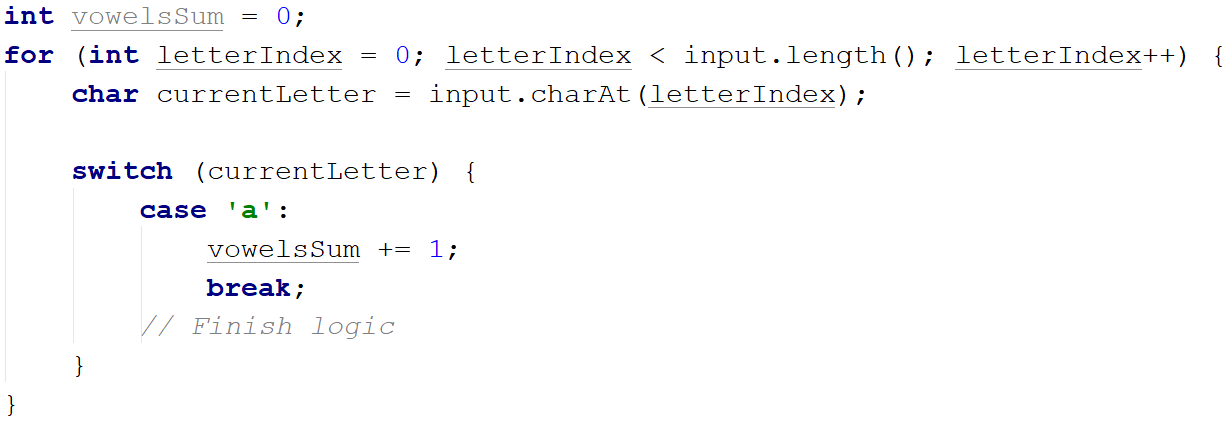
1. Прочетете входния текст input:



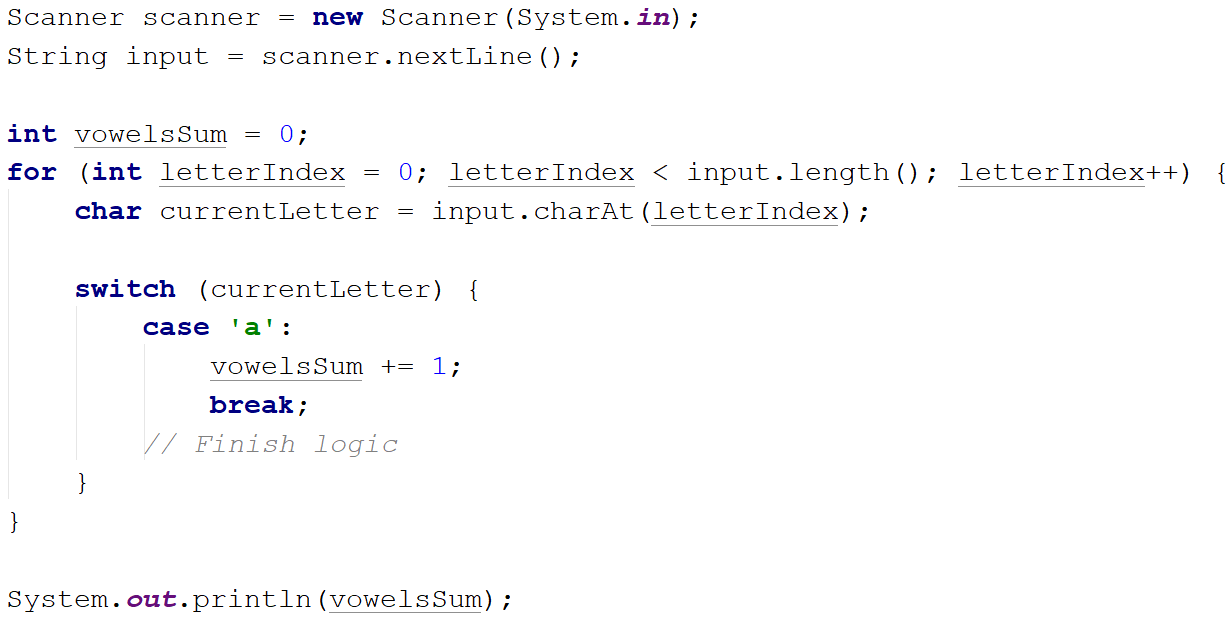
1. Направете for цикъл с начална стойност на контролната променлива от 0 до input.Length (дължината на текста). На всяка итерация взимайте буквата на позиция във въведената дума равна на стойността на контролната променлива letterIndex, чрез метода charAt()



1. Направете switch конструкция, в която да проверите стойностите на буквата, според условието на задачата. Инициализирайте променлива vowelsSum, в която да пазите сумата от проверените букви:



1. Принтирайте стойността на променливата vowelsSum



**Примерна изпитна задача**

## Умната Лили

Лили вече е на n **години**. За всеки свой **рожден ден** тя получава подарък. За **нечетните** рождени дни (**1, 3, 5...** n) получава **играчки**, а за всеки **четен** (**2, 4, 6...** n) получава **пари**. За **втория рожден ден** получава **10.00 лв**, като **сумата се увеличава с 10.00 лв, за всеки следващ четен рожден ден** (**2 -> 10**, **4 -> 20**, **6 -> 30**...и т.н.). През годините Лили тайно е спестявала парите. **Братът** на Лили, **в годините**, които тя **получава пари**, **взима по 1.00 лев** от тях. Лили **продала играчките** получени през годините, **всяка за P лева** и **добавила** сумата **към спестените пари**. С парите искала да си **купи пералня за X лева**. Напишете програма, която да пресмята, **колко пари е събрала** и дали ѝ **стигат да си купи пералня**.

### Вход

Програмата прочита **3 числа**, въведени от потребителя, на отделни редове:

* **Възрастта** на Лили - **цяло число** в интервала **[1...77]**
* **Цената на пералнята** - число в интервала **[1.00...10 000.00]**
* **Единична цена** **на играчка** - **цяло** **число** в интервала **[0...40]**

### Изход

Да се отпечата на конзолата един ред:

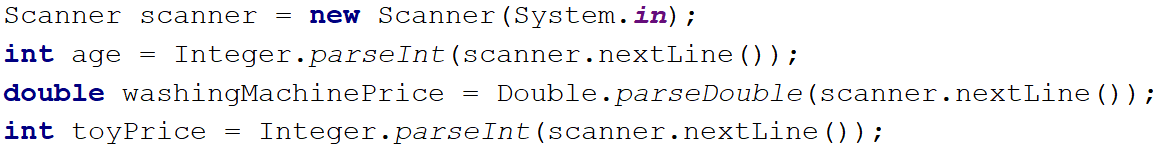
* Ако парите на Лили са достатъчни:
  + **“**Yes! {N}**”** - където **N** е остатъка пари след покупката
* Ако парите не са достатъчни:
  + **“No! {М}“** - където **M** е сумата, която не достига
* Числата **N** и **M** трябва да за **форматирани до вторият знак след десетичната запетая**.

### Примерен вход и изход

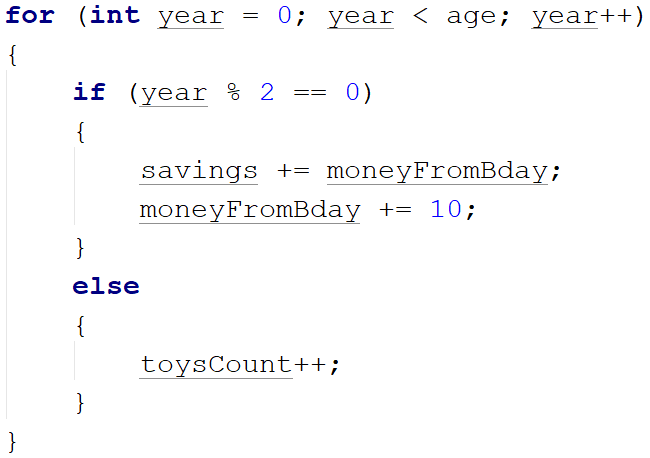
|  |  |  |
| --- | --- | --- |
| **вход** | **изход** | **Коментари** |
| 10  170.00  6 | Yes! 5.00 | **Първи рожден ден** получава **играчка**; **2ри -> 10лв**; 3ти -> играчка;  **4ти** -> 10 + 10 = **20лв**; **5ти** -> играчка; **6ти** -> 20 + 10 = **30лв**; **7ми** -> играчка;  **8ми** -> 30 + 10 = **40лв**; **9ти** -> играчка; **10ти** -> 40 + 10 = **50лв**.  **Спестила е** -> 10 + 20 + 30 + 40 + 50 = **150лв**. Продала е **5 играчки по 6 лв** = **30лв**.  **Брат ѝ взел 5 пъти по 1 лев** = **5лв**. **Остават** -> 150 + 30 – 5 = **175лв**.  **175 >= 170** (цената на пералнята) **успяла** е да я купи и са и **останали** 175-170 = **5 лв.** |
| 21  1570.98  3 | No! 997.98 | **Спестила е 550лв**. **Продала** е **11** играчки по **3 лв** = **33лв**. Брат ѝ **взимал 10** години по **1 лев** = **10лв**. **Останали** 550 + 33 – 10 = **573лв**  **573 < 1570.98** – **не е успяла** да купи пералня. **Не ѝ достигат** 1570.98–573 = **997.98лв** |

### Насоки

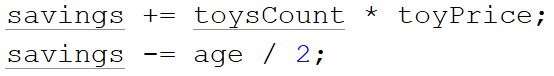
1. Прочетете **входните данни - възрастта** на Лили, **цената на пералнята** и **цената на една играчка**:



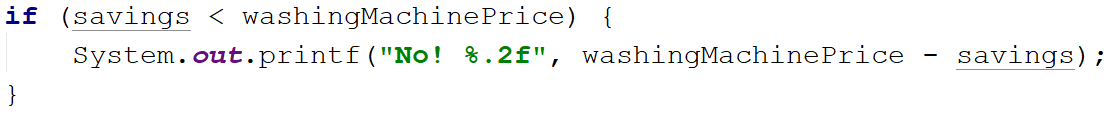
1. Във for цикъл според годината, на която е рождения ден на Лили, добавяйте към спестяванията й, или към броя играчки. Създайте променлива savings и moneyFromBday, с начална стойност **0** и **10**, в които да пазите стойността на спестяванията и парите, които Лили трябва да получи на всеки рожден ден:



1. Добавете към спестяванията на Лили печалбата от продажбата на играчките. След това извадете сумата, която брат й е вземал през годините:



1. Принтирайте изхода според условието на задачата:
   1. Ако спестяванията на Лили **не достигат** за купуването на пералнята, принтирайте:



* 1. Ако спестяванията на Лили **достигат** за купуването на пералнята, принтирайте:

