# Упражнения: Повторения (цикли)

Тествайте решенията си в **Judge системата**: <https://judge.softuni.org/Contests/Practice/Index/3898>

Състояние на настоящия учебен материал:

A green and blue rectangle with white text

Description automatically generated

## Числа от 1 до 100

Напишете програма, която отпечатва числата **от 1 до 100**, всяко на нов ред.

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| (няма) | 1  2  3  …  98  99  100 |

### Насоки

1. Създайте **нов проект с име** "Numbers1To100".
2. Създайте метода Main(String[] args):

A picture containing logo

Description automatically generated

1. Създайте **for-цикъл**, който да започва от **1** и да завършва на **10**:

A picture containing chart

Description automatically generated

1. За всяка итерация на for-цикъла отпечатайте **текущото число**:

A picture containing company name

Description automatically generated

## Числата от N до 1 в обратен ред

Напишете програма, която чете цяло положително число n, въведено от потребителя, и отпечатва **числата от** n **до** **1** **в обратен ред** (от най-голямото към най-малкото).

### Примери

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Вход** | **Изход** |  | **Вход** | **Изход** |  | **Вход** | **Изход** |
| 2 | 2  1 | 3 | 3  2  1 | 5 | 5  4  3  2  1 |

### Насоки

1. Прочетете входните данни – числото **n**:

Text

Description automatically generated with low confidence

1. Създайте **for-цикъл**, който да итерира от **n** до **1**:

Chart

Description automatically generated with low confidence

1. За всяка итерация на **for-цикъла** отпечатайте **текущото число**:

A picture containing text

Description automatically generated

## Числата от 1 до N през 3

Напишете програма, която чете число n, въведено от потребителя, и отпечатва **числата от 1 до** n **през 3** (със стъпка 3).

### Примери

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Вход** | **Изход** |  | **Вход** | **Изход** |  | **Вход** | **Изход** |
| 10 | 1  4  7  10 | 7 | 1  4  7 | 15 | 1  4  7  10  13 |

### Насоки

1. Прочетете входните данни – **числото n**:

Text

Description automatically generated with medium confidence

1. Създайте **for-цикъл**, който итерира **от 1 до n** със **стъпка 3**:

A picture containing chart

Description automatically generated

1. За всяка итерация на **for-цикъла** отпечатайте текущото число.

## Поток от символи

Напишете програма, която чете **текст** (стринг), въведен от потребителя, и печата **всеки символ от текста** на отделен ред.

### Примери

|  |  |  |  |
| --- | --- | --- | --- |
| **Вход** | **Изход** | **Вход** | **Изход** |
| softuni | s  o  f  t  u  n  i | ice cream | i  c  e    c  r  e  a  m |

### Насоки

1. Прочетете входните данни – въведения от потребителя **текст**:

A picture containing diagram

Description automatically generated

1. Направете **for-цикъл**, който започва от **0** и итерира до **дължината на масива**:

A picture containing chart

Description automatically generated

1. На всяка итерация на **for-цикъла** взимайте **буквата** на позиция във въведената дума равна на стойността на **контролната променлива** i:

A picture containing chart

Description automatically generated

1. На всяка итерация принтирайте стойността на променливата **letter**.

## 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**.

### Примери

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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 входните данни - числото **n**:

Text

Description automatically generated with medium confidence

1. Създайте променлива **sum**, която да съхранява **сумата от числата**, които ще се прочетат от конзолата:

Icon

Description automatically generated

1. Създайте **for-цикъл**, в който да вземете **n** на брой числа и да ги **добавите към сумата**:

Text

Description automatically generated with low confidence

1. След цикъла **отпечатайте** **сумата** на конзолата.

## Число, кратно на 10

Напишете програма, която чете **цяло** **число** n, въведено от потребителя. При въвеждане на число, което **не се дели на 10 без остатък**, да се отпечата съобщение за **грешка** и потребителят да се подкани **да въведе ново число**, докато не въведе **подходящо число**.

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 40 | The number is: 40 |
| **105**  **101**  **-202**  -20 | Invalid number!  Invalid number!  Invalid number!  The number is: -20 |

### Насоки

1. Прочетете входните данни – **числото n**:

Graphical user interface, text

Description automatically generated with medium confidence

1. Създайте **while-цикъл**, с който да итерирате докато не получите число, което се **дели на 10** **без остатък**:

A picture containing text

Description automatically generated

1. След while-цикъла **отпечатайте очаквания изход**.

## Четене на думи

Напишете програма, която чете **текст** (стринг) от конзолата и го принтира, докато не получи **командата** "**Stop**".

### Примери

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Вход** | **Изход** |  | **Вход** | **Изход** |
| Nakov  SoftUni  Sofia  Bulgaria  SomeText  Stop  AfterStop  Europe  HelloWorld | Nakov  SoftUni  Sofia  Bulgaria  SomeText |  | Sofia  Berlin  Moscow Athens  Madrid  London  Paris  Stop  AfterStop | Sofia  Berlin  Moscow Athens  Madrid  London  Paris |

### Насоки

1. Прочетете **входния текст**:

A picture containing diagram

Description automatically generated

1. Създайте **while-цикъл**, с който да итерирате до получаване на **команда "Stop"**:

A picture containing chart

Description automatically generated

1. Отпечатайте **текущата дума** и прочетете от конзолата **нова дума**:

Text

Description automatically generated with medium confidence

## Сумиране на гласните букви

Да се напише програма, която чете **текст** (стринг), въведен от потребителя, и 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 |

### Примери

|  |  |  |
| --- | --- | --- |
| **Вход** | **Изход** | **Обяснения** |
| hello | 6 | e + o = 2 + 4 = 6 |
| Hi | 3 | i = 3 |
| bamboo | 9 | a + o + o = 1 + 4 + 4 = 9 |

### Насоки

1. Прочетете входните данни – **текста**, въведен от потребителя:

A picture containing graphical user interface

Description automatically generated

1. Създайте променлива, в която да съхранявате **сумата от стойностите на гласните букви**:

Icon

Description automatically generated with low confidence

1. Създайте **for-цикъл**, който да итерира през **всеки символ** на въведения текст:

Graphical user interface

Description automatically generated with low confidence

1. Създайте поредица от **if-else if проверки**, която **добавя съответната стойност** за всяка различна **гласна** буква към **сумата**:

Graphical user interface, text

Description automatically generated with medium confidence

1. След цикъла отпечатайте **сумата от стойностите на гласните букви**.

## Сума от числа

Напишете програма, която чете **цяло число** от конзолата и на **всеки следващ ред цели числа**, докато тяхната сума стане **по-голяма или равна** на **първоначалното число**. След приключване да се отпечата **сумата на въведените числа**.

### Примери

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Вход** | **Изход** | **Коментар** |  | **Вход** | **Изход** | **Коментар** |
| **100**  10  20  30  40 | 100 | 10+20+30+40 = 100 |  | **20**  1  2  3  4  5  6 | 21 | 1+2+3+4+5+6 = 21 |

### Насоки

1. Прочетете от конзолата цялото число **n**.
2. Създайте променлива, която да съхранява **сумата** от въведените числа.
3. Създайте цикъл, с който да итерирате докато **сумата е по-малка от n**.
4. За всяка итерация на цикъла прочетете **ново цяло число** и го **добавете към сумата**.
5. След цикъла отпечатайте **сумата от числата**.

## Парола

Напишете програма, която първоначално прочита **име** и **парола** на потребителски профил. След това чете **парола за вход**, при въвеждане на **грешна парола**, потребителя да се подкани да въведе **нова парола**.

### Примери

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Вход** | **Изход** |  | **Вход** | **Изход** |
| Nakov  1234  pass  1324  1234 | Welcome Nakov! | Gosho  secret  secret | Welcome Gosho! |

### Насоки

1. Инициализирайте две променливи username и password, които ще съдържат потребителското име и паролата.
2. Инициализирайте променлива input, която ще държи въведената от потребителя парола за вход.
3. В while**-цикъл**, до въвеждане на валидна парола, четете нова парола.
4. Когато се въведе **валидна парола**, **принтирайте съобщението за успешен вход**.

## Най-голямо число

Напишете програма, която до получаване на **командата "Stop"** чете **цели числа**, въведени от потребителя, и намира **най-голямото** измежду тях. Въвеждат се по **едно число на ред**.

### Примери

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Вход** | **Изход** |  | **Вход** | **Изход** |  | **Вход** | **Изход** |  | **Вход** | **Изход** |  | **Вход** | **Изход** |
| 100  99  80  70  Stop | 100 | -10  20  -30  Stop | 20 | 45  -20  7  99  Stop | 99 | 999  Stop | 999 | -1  -2  Stop | -1 |

### Насоки

1. Създайте променлива, която да съхранява **най-голямото число** и има начална стойност **int.MinValue**.
2. Прочетете **входните данни**.
3. Създайте **while-цикъл**, с който да итерирате до получаване на команда "**Stop**".
4. Проверете дали прочетеното число е **по-голямо** от текущото най-голямо число.
   * Ако е **по-голямо**, запазете това число като **максимално**
5. Преминете към **следващите входни данни**.

## Най-малко число

Напишете програма, която до получаване на **командата "Stop"** чете **цели числа**, въведени от потребителя, и намира **най-малкото** измежду тях. Въвежда се по едно число на ред.

### Примери

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Вход** | **Изход** |  | **Вход** | **Изход** |  | **Вход** | **Изход** |  | **Вход** | **Изход** |  | **Вход** | **Изход** |
| 100  99  80  70  Stop | 70 | -10  20  -30  Stop | -30 | 45  -20  7  99  Stop | -20 | 999  Stop | 999 | -1  -2  Stop | -2 |

## Редица цели числа

Напишете програма, която чете **n на брой цели числа**. Принтирайте **най-голямото** и **най-малкото** число измежду въведените, както е показано в примера по-долу.

На **първия ред** от входа е дадено числото **n**, a на **следващите n реда** е дадено по едно **цяло число**.

### Примери

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Вход** | **Изход** |  | **Вход** | **Изход** |
| **5**  10  20  304  0  50 | Max number: 304  Min number: 0 | **6**  250  5  2  0  100  1000 | Max number: 1000  Min number: 0 |

## Часовник

Напишете програма, която отпечатва **часовете в денонощието от 0:0 до 23:59**, всеки на отделен ред. Часовете трябва да се изписват във формат **"{час}:{минути}"**.

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| *(няма вход)* | 0:0  0:1  0:2  0:3  0:4  0:5  0:6  0:7  0:8  0:9  0:10  ...  23:50  23:51  23:52  23:53  23:54  23:55  23:56  23:57  23:58  23:59 |

## Таблица за умножение

Отпечатайте на конзолата **таблицата за умножение** за числата **от 1 до 10** във формат:   
**"{първи множител} \* {втори множител} = {резултат}"**.

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| *(няма вход)* | 1 \* 1 = 1  1 \* 2 = 2  1 \* 3 = 3  1 \* 4 = 4  1 \* 5 = 5  1 \* 6 = 6  1 \* 7 = 7  1 \* 8 = 8  1 \* 9 = 9  1 \* 10 = 10  ...  10 \* 1 = 10  10 \* 2 = 20  10 \* 3 = 30  10 \* 4 = 40  10 \* 5 = 50  10 \* 6 = 60  10 \* 7 = 70  10 \* 8 = 80  10 \* 9 = 90  10 \* 10 = 100 |