# Упражнения: Хващане на изключения

Тествайте решението в Judge: <https://judge.softuni.org/Contests/4168/09-Exception-Handling>.

## Корен квадратен

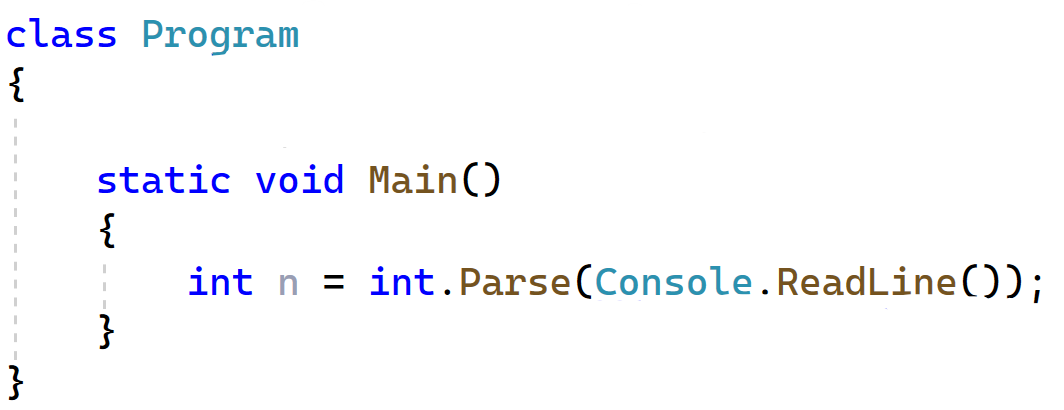
Напишете програма, която чете число и отпечатва неговия **квадрат**. Ако числото е **отрицателно**, отпечатайте "**Invalid number.**". Накрая отпечатайте "**Good bye.**". Използвайте **try-catch-finally.**

### Примери

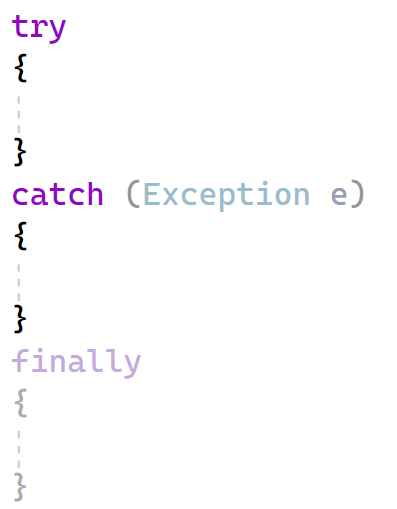
|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 9 | 3  Good bye. |
| -1 | Invalid number.  Good bye. |

### Насоки

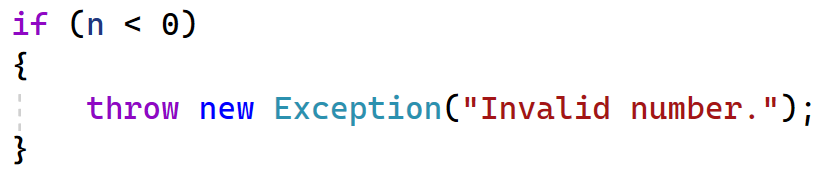
1. Първо прочетете **числото**:



1. Създайте **try-catch-finally** блок:



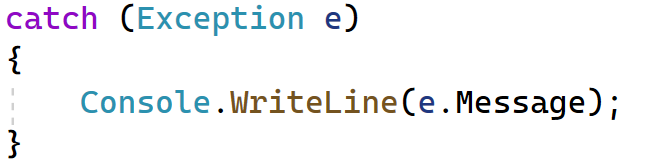
1. В **try** **блока** сложете проверка дали **n** е **по-малко** от **нула**. Ако е така, хвърлете **exception**:



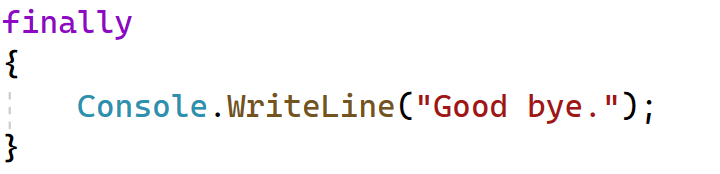
1. След проверката отпечатайте **квадрата** на **числото**:



1. В **catch** **блока** отпечатайте съобщението на хванатия **exception**:



1. Накрая в **finally** **блока** отпечатайте съобщението **"Good bye."**:



## Сума от числа

Ще получите редица от елементи от **различен тип**, разделени с **интервал**. Вашата задача е да пресметнете **сумата на всички валидни числа** от входа. Опитайте се да добавите всеки един елемент от масива към сумата. Напишете **съобщение** за всяко възможно **изключение,** докато правите операции с елементите:

* Ако получите елемент, който **не е** в **правилния формат** **(FormatException)**, отпечатайте:  
  **"The element '{елемент}' is in wrong format!"**
* Ако получите елемент, който **не е обхвата** на **integer (OverflowException),** отпечатайте:  
  **"The element '{елемент}' is out of range!"**

След всеки опит за добавяме на елемент отпечатайте:

**"Element '{елемент}' processed - current sum: {сума}"**

Накрая отпечатайте **сумата от всички числа**:

**"The total sum of all integers is: {сума}"**

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 2147483649 2 3.4 5 invalid 24 -4 | The element '2147483649' is out of range!  Element '2147483649' processed - current sum: 0  Element '2' processed - current sum: 2  The element '3.4' is in wrong format!  Element '3.4' processed - current sum: 2  Element '5' processed - current sum: 7  The element 'invalid' is in wrong format!  Element 'invalid' processed - current sum: 7  Element '24' processed - current sum: 31  Element '-4' processed - current sum: 27  The total sum of all integers is: 27 |
| 9876 string 10 -2147483649 -8 3 4.86555 8 | Element '9876' processed - current sum: 9876  The element 'string' is in wrong format!  Element 'string' processed - current sum: 9876  Element '10' processed - current sum: 9886  The element '-2147483649' is out of range!  Element '-2147483649' processed - current sum: 9886  Element '-8' processed - current sum: 9878  Element '3' processed - current sum: 9881 The element '4.86555' is in wrong format!  Element '4.86555' processed - current sum: 9881  Element '8' processed - current sum: 9889  The total sum of all integers is: 9889 |

### Насоки

1. Прочетете **елементите** като **низове**.
2. Създайте променлива, в която ще съхранявате **сумата**.
3. Създайте **foreach-цикъл**, който ще обхожда всички елементи от входа.
4. Създайте **try** **блок**.
5. В него се опитайте да добавите към сумата елемента.
6. Създайте **catch блок**, който ще хваща изключения от тип **FormatException** и ще отпечатва "**The element '{елемент}' is in wrong format!**".
7. Създайте и още един **catch**, който ще хваща изключения от тип **OverflowException** и ще отпечатва **"The element '{елемент}' is out of range!"**.
8. След това създайте **finally** **блок**, който ще отпечатва "**Element '{елемент}' processed - current sum: {сума}**".
9. Накрая отпечатайте **сумата** на **елементите**.

## Въвеждане на числа

Напишете метод **ReadNumber(int start, int end),** който получава число и проверява дали е в интервала (**start…end**) (не броим самите **start** и **end**). Ако числото е **невалидно** или е **текст**, методът трябва да хвърли **изключение.**

Напишете програма, която чете **10 числа** в диапазона между **1** и **100** чрез метода **ReadNumber**. Ако потребителят въведе **грешно** число, **игнорирайте** го. Продължавайте да въвеждате, докато не станат **10 валидни числа**. Отпечатайте елементите на списъка с валидните числа, разделени със **запетая и интервал**.

### Примери

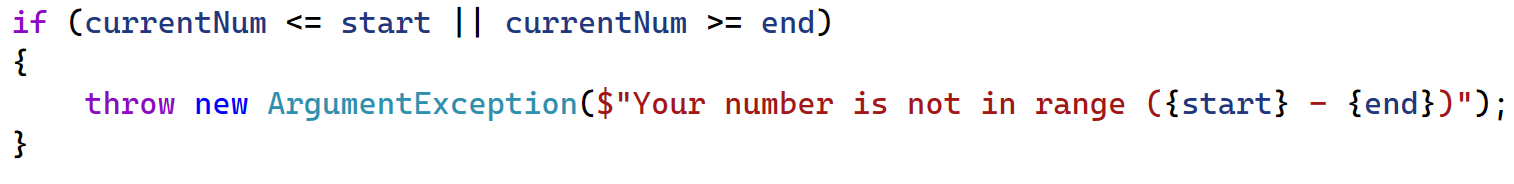
|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 2  3  4  5  6  7  8  9  10  11 | 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 |
| 1  2  1  3  p  4  5  6  7  8  9  10  11 | Your number is not in range (1 - 100)  Your number is not in range (1 – 100)  Invalid Number!  2, 3, 4, 5, 6, 7, 8, 9, 10, 11 |

### Насоки

1. Започнете от създаването на **метода** **ReadNumber(int start, int end)**:
2. Добавете **try-catch** блок. **catch блок-ът** ще служи за невалидни подадени данни (текст например):
3. В **try** **блока** ще прочетете числото:



1. След това създайте проверка дали **числото** е **по-малко** или **равно** на **start** или е **по-голямо** или **равно** на **end**. Ако е така, хвърлете **изключение** със **съобщение** **"Your number is not in range {start} - {end}"**:

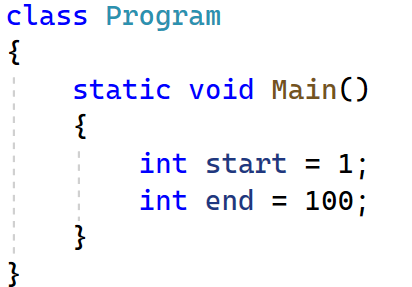


1. Накрая върнете **числото**.
2. В **catch** **блок**-а хванете изключението при въведен текст и го хвърлете наново (за да бъде handle-ното от Main() метода:

A close up of a word

Description automatically generated

1. Сега се върнете **Main()** метода и създайте две променливи – **start** със **стойност 1** и **end** със **стойност 100**:



1. Създайте **списък**, в който да съхранява валидните числа:



1. След това създайте **while-цикъл**, който итерира докато **броя** на **елементите** на **списъка** е **по-малък** от **10**:

A close-up of a website

Description automatically generated

1. За всяка итерация на **while-цикъла** извикайте метода **ReadNumber()** и добавете **резултата** от метода към **validNums** (ако всичко с метода е било наред и числото е валидно):

A close up of text

Description automatically generated

1. В случай на **изключение** отпечатайте неговото **съобщение**:

A close up of words

Description automatically generated

1. Накрая отпечатайте **валидните числа**:



## Карти

Създайте клас **Card, който съдържа стойност и боя.**

* **Валидни стойности са: 2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K, A**
* **Валидни бои са: S (**♠), H (♥), D (♦), C (♣)

Стойността и боята ще бъдат с **главна** буква. Ще трябва да използвате метода toString(), който отпечатва стойността и боята на картата във формата:

**"[{стойност}{боя}]" – пример: [A♠] [5♣] [10♦]**

Използвайте следните UTF кодове за боите:

* **\u2660** – Spades (♠)
* **\u2665** – Hearts (♥)
* **\u2666** – Diamonds (♦)
* **\u2663** – Clubs (♣)

Напишете програма, която взима тесте карти като масив от низове. Отпечатайте ги като редица от карти, разделени с **интервал**. Отпечатайте **"Invalid card!",** когато има **невалидна** карта във входа.

### Вход

* Ще получите стойностите и боите на картите на един ред във формата:

**"{стойност} {боя}, {стойност} {боя}, …"**

### Изход

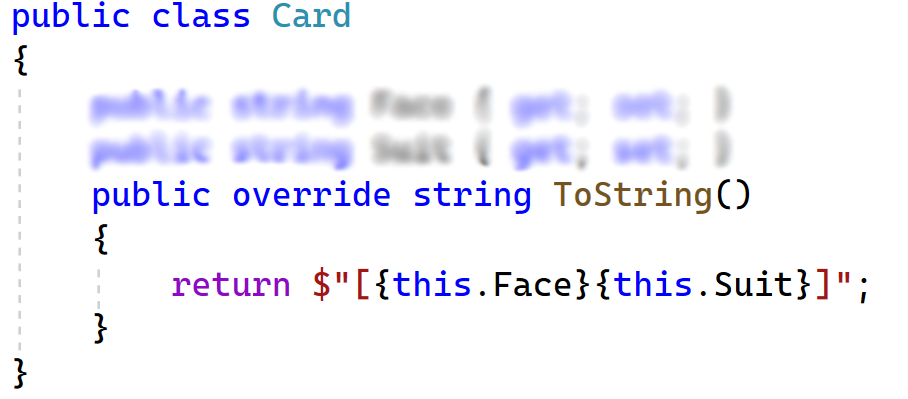
* Отпечатайте на конзолата масива от карти като низ, разделени с интервал.

### Примери

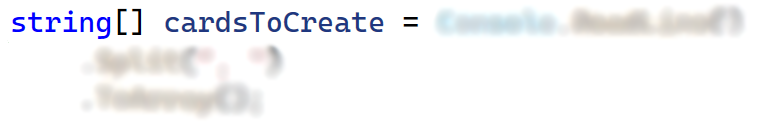
|  |  |
| --- | --- |
| **Вход** | **Изход** |
| A S, 10 D, K H, 2 C | [A♠] [10♦] [K♥] [2♣] |
| 5 C, 10 D, king C, K C, Q heart, Q H | Invalid card!  Invalid card!  [5♣] [10♦] [K♣] [Q♥] |

### Насоки

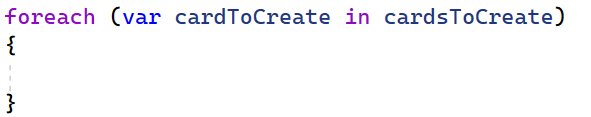
1. След **namespace-a** създайте клас **Card** със свойства **Face**, **Suit**. Използвайте метода toString(), който трябва да връща "[**{Стойност}{Боя}**]":



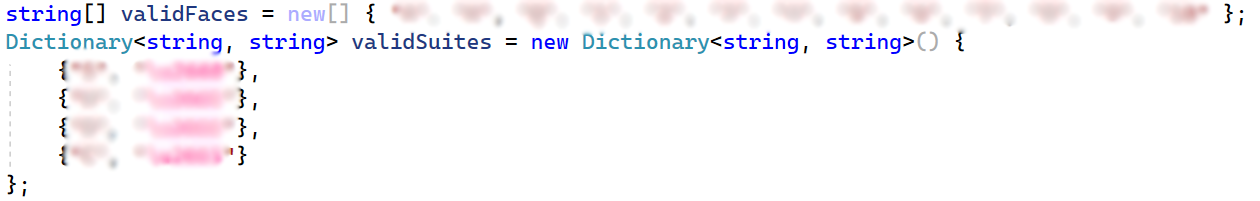
1. В **Main()** прочетете картите:



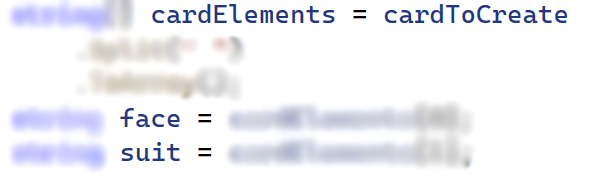
1. След това създайте **списък**, който да съхранява картите.
2. Създайте **foreach-цикъл**, който да обхожда картите прочетени от входа:



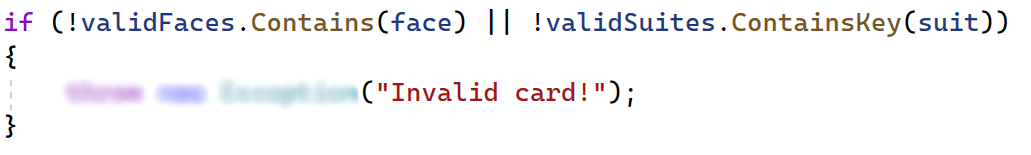
1. В **цикъла** създайте **променлива** **validFaces,** която е **масив** от **низове** и съдържа всички **валидните стойности на картите,** и **променлива** **validSuites**, която е **речник** и съдържа **валидните** **бои**:



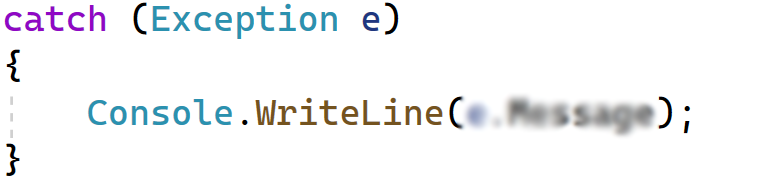
1. След това разделете низа за да получите **стойността** и **боята** на **картата**:



1. Създайте **try-catch блок**.
2. В **try** **блока** проверете дали **картата** съдържа **невалидна** **стойност** или **невалидна** **боя**. Ако е така, **хвърлете изключение** със съобщение "**Invalid card!**":



1. След това създайте **клас** **Card**, с данните **прочетени** от **входа** и го добавете в списъка.
2. В **catch** **блока** отпечатайте съобщението на изключението:



1. Накрая отпечатайте **картите**.

## Игра с Catch

Ще получите на първия ред масив от числа. След това ще получите команди, с които трябва да манипулирате масива:

* **"Replace {индекс} {елемент}"** – заменя елемента на дадения индекс с **нова стойност**.
* **"Print {старт} {край}"** – отпечатва **елементите** от **старта** до **края** включително.
* **"Show {индекс}"** – отпечатва **елемента** на дадения **индекс**.

Вашата задача е да напишете съобщения за **изключения**,които могат да се случат в **програмата**:

* Ако получите **индекс**, който **не е валиден** в масива, отпечатайте:  
  **"**The index does not exist!**"**
* Ако получите **променлива**, която **не е валидна**, отпечатайте:  
  **"**The variable is not in the correct format!**"**

Когато хванете **3 изключения**, спрете входа и отпечатайте елементите в масива, разделени с **", "**.

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| **1 2 3 4 5**  Replace 1 9  Replace 6 3  Show 3  Show pesho  Show 6 | The index does not exist!  4  The variable is not in the correct format!  The index does not exist!  1, 9, 3, 4, 5 |
| 1 2 3 4 5  Replace 3 9  Print 1 4  Print -3 12  Print 1 5  Show 3  Show 12.3 | 2, 3, 9, 5  The index does not exist!  The index does not exist!  9  The variable is not in the correct format!  1, 2, 3, 9, 5 |

## Транзакция на пари

Ще получите на първия ред колекция от банкови профили, съдържащи **номер на профила** (**integer**) и   
**баланс** (**double**) във формата:

**"{номер на профила}-{баланс},{номер на профила}-{баланс},…"**

Докато не получите **"End",** ще получавате команди за манипулиране на баланс:

* **"Deposit {номер на профила} {сума}"** – **добавя сума** към баланса на профила.
* **"Withdraw {номер на профила} {сума}"** – **намалява** баланса на профила със дадената **сума**.

Отпечатайте **съобщения** за следните **изключения**, които могат да се случат в **програмата:**

* Ако получите **невалидна команда**, отпечатайте:  
  **"Invalid command!"**
* Ако получите акаунт, който **не съществува**, отпечатайте:  
  **"**Invalid account!**"**
* Ако получите "**Withdraw**" и сумата **е по-голяма** от баланса, отпечатайте:  
  **"**Insufficient balance!**"**

В случай че командата е **правилна**, отпечатайте:

**"Enter another command"**

След всяка **успешна операция** отпечатайте **профила**. Форматирайте баланса до **втората цифра** след десетичната запетая.

**"Account {номер на профила} has new balance: {баланс}"**

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 1-45.67,2-3256.09,3-97.34  Deposit 1 50  Withdraw 3 100  End | Account 1 has new balance: 95.67  Enter another command  Insufficient balance!  Enter another command |
| 1473653-97.34,44643345-2347.90  Withdraw 1473653 150.50  Deposit 44643345 200  Block 1473653 30  Deposit 1 30  End | Insufficient balance!  Enter another command  Account 44643345 has new balance: 2547.90  Enter another command  Invalid command!  Enter another command  Invalid account!  Enter another command |