Исключения в программировании и их обработка. Обучение в записи

Задание 1. Преобразование строки в число

Реализуйте метод `convertAndSum`, который принимает массив строк, каждая из которых должна быть преобразована в целое число. Метод возвращает сумму всех чисел. Если хотя бы одна строка не может быть преобразована в число, метод должен выбросить исключение `NumberFormatException`. Дополнительно, если сумма чисел превышает 100, выбрасывайте `ArithmeticException` с сообщением "Превышен лимит суммы".

```
class Answer {

    public static int convertAndSum(String[] strings) {

        // Введите свое решение ниже

    }

}


public class Printer {

    public static void main(String[] args) {

        // Первая попытка: корректный ввод

        try {

            String[] strings = {"10", "20", "70"};

            System.out.println(Answer.convertAndSum(strings));
// Ожидаемый результат: 100

        } catch (NumberFormatException e) {

            System.out.println("Ошибка преобразования строки в
число");

        } catch (ArithmeticException e) {
```

```
            System.out.println(e.getMessage());
        }

        // Вторая попытка: ввод с некорректным числом
        try {
            String[] invalidStrings = {"10", "20", "abc"};

System.out.println(Answer.convertAndSum(invalidStrings));  //
Ожидаемый результат: исключение NumberFormatException
        } catch (NumberFormatException e) {
            System.out.println("Ошибка преобразования строки в
число");
        } catch (ArithmeticException e) {
            System.out.println(e.getMessage());
        }

        // Третья попытка: сумма превышает лимит
        try {
            String[] overLimitStrings = {"50", "60"};

System.out.println(Answer.convertAndSum(overLimitStrings));
// Ожидаемый результат: исключение ArithmeticException
        } catch (NumberFormatException e) {
            System.out.println("Ошибка преобразования строки в
число");
        } catch (ArithmeticException e) {
            System.out.println(e.getMessage());
```

```
        }

    }

}
```

Подсказка № 1

Используйте цикл `for` для перебора всех элементов массива строк. Для каждой строки используйте метод `Integer.parseInt` для преобразования строки в целое число. Это может вызвать `NumberFormatException`, если строка не может быть преобразована.

Подсказка № 2

Обработайте исключение `NumberFormatException`, чтобы программа не прерывалась при обнаружении некорректного ввода. В случае возникновения исключения, выбросите `NumberFormatException` в вызывающий метод, чтобы информировать о проблеме.

Подсказка № 3

Создайте переменную для хранения суммы чисел. Внутри цикла добавляйте каждое корректно преобразованное число к этой переменной.

Подсказка № 4

После окончания цикла, проверьте, не превышает ли сумма чисел 100. Если превышает, выбросьте `ArithmeticException` с соответствующим сообщением.

Эталонное решение:

```
class Answer {

   public static int convertAndSum(String[] strings) {

        int sum = 0;

        for (String s : strings) {

            try {

                int number = Integer.parseInt(s);

                sum += number;

            } catch (NumberFormatException e) {
```

```
                // Перехватываем ошибку преобразования и выбрасываем
снова
                throw new NumberFormatException("Ошибка
преобразования строки в число: " + s);
            }
        }
        if (sum > 100) {
            throw new ArithmeticException("Превышен лимит суммы");
        }
        return sum;
    }
}


public class Printer {
    public static void main(String[] args) {
        // Первая попытка: корректный ввод
        try {
            String[] strings = {"10", "20", "70"};
            System.out.println(Answer.convertAndSum(strings));  //
Ожидаемый результат: 100
        } catch (NumberFormatException e) {
            System.out.println(e.getMessage());
        } catch (ArithmeticException e) {
            System.out.println(e.getMessage());
        }

        // Вторая попытка: ввод с некорректным числом
        try {
```

```
            String[] invalidStrings = {"10", "20", "abc"};

            System.out.println(Answer.convertAndSum(invalidStrings));
// Ожидаемый результат: исключение NumberFormatException

        } catch (NumberFormatException e) {

            System.out.println(e.getMessage());

        } catch (ArithmeticException e) {

            System.out.println(e.getMessage());

        }



        // Третья попытка: сумма превышает лимит

        try {

            String[] overLimitStrings = {"50", "60"};


System.out.println(Answer.convertAndSum(overLimitStrings));  //
Ожидаемый результат: исключение ArithmeticException

        } catch (NumberFormatException e) {

            System.out.println(e.getMessage());

        } catch (ArithmeticException e) {

            System.out.println(e.getMessage());

        }

    }

}
```

Задача 2. Объединение массивов с проверкой длины и содержимого

Реализуйте метод `mergeAndValidateArrays`, который принимает два массива целых чисел. Метод должен объединить массивы и вернуть новый массив. Если длины массивов не равны, выбрасывайте исключение `IllegalArgumentException`. Если хотя бы один элемент объединенного

массива отрицательный, выбрасывайте исключение `RuntimeException` с сообщением "Обнаружен отрицательный элемент".

```
import java.util.Arrays;


class Answer {

    public static int[] mergeAndValidateArrays(int[] a, int[]
b) {

        // Введите свое решение ниже


    }

}


public class Printer {

    public static void main(String[] args) {

        try {

            int[] a = {1, 2, 3};

            int[] b = {4, 5, 6};

            int[] result = Answer.mergeAndValidateArrays(a, b);

            System.out.println(Arrays.toString(result));  //
Ожидаемый результат: [1, 2, 3, 4, 5, 6]

        } catch (IllegalArgumentException e) {

            System.out.println("Длины массивов не равны");

        } catch (RuntimeException e) {

            System.out.println(e.getMessage());

        }

        try {
```

```
            int[] c = {1, 2};
            int[] d = {3, 4, 5};

System.out.println(Arrays.toString(Answer.mergeAndValidateArra
ys(c, d)));  // Ожидаемый результат: исключение
IllegalArgumentException
        } catch (IllegalArgumentException e) {
            System.out.println("Длины массивов не равны");
        } catch (RuntimeException e) {
            System.out.println(e.getMessage());
        }
        try {
            int[] e = {-1, 2, 3};
            int[] f = {4, 5, 6};

System.out.println(Arrays.toString(Answer.mergeAndValidateArra
ys(e, f)));  // Ожидаемый результат: исключение
RuntimeException
        } catch (IllegalArgumentException e) {
            System.out.println("Длины массивов не равны");
        } catch (RuntimeException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

Подсказка № 1

Перед тем как объединять массивы, проверьте их длины. Используйте условие `if` для сравнения длины двух массивов. Если длины не равны, выбросьте исключение `IllegalArgumentException` с соответствующим сообщением.

Подсказка № 2

Для объединения двух массивов используйте метод `System.arraycopy` или простое объединение с помощью цикла `for`. Создайте новый массив, который будет вдвое длиннее исходных массивов, и скопируйте в него элементы обоих массивов.

Подсказка № 3

После объединения массивов пройдитесь по элементам объединенного массива с помощью цикла `for`. Если обнаружите отрицательное значение, выбросьте исключение `RuntimeException` с сообщением "Обнаружен отрицательный элемент".

Подсказка № 4

Используйте метод `System.arraycopy` для эффективного копирования элементов из одного массива в другой. Это поможет избежать сложностей при ручном копировании элементов.

Эталонное решение:

```
import java.util.Arrays;


class Answer {

   public static int[] mergeAndValidateArrays(int[] a, int[] b) {

       if (a.length != b.length) {

           throw new IllegalArgumentException("Длины массивов не
равны");

       }


       int[] mergedArray = new int[a.length + b.length];

       System.arraycopy(a, 0, mergedArray, 0, a.length);

       System.arraycopy(b, 0, mergedArray, a.length, b.length);
```

```
        for (int num : mergedArray) {
            if (num < 0) {
                throw new RuntimeException("Обнаружен отрицательный
элемент");
            }
        }

        return mergedArray;
    }
}

public class Printer {
    public static void main(String[] args) {
        try {
            int[] a = {1, 2, 3};
            int[] b = {4, 5, 6};
            int[] result = Answer.mergeAndValidateArrays(a, b);
            System.out.println(Arrays.toString(result));  //
Ожидаемый результат: [1, 2, 3, 4, 5, 6]
        } catch (IllegalArgumentException e) {
            System.out.println("Длины массивов не равны");
        } catch (RuntimeException e) {
            System.out.println(e.getMessage());
        }
        try {
            int[] c = {1, 2};
            int[] d = {3, 4, 5};
```

```
System.out.println(Arrays.toString(Answer.mergeAndValidateArrays(c,
d)));  // Ожидаемый результат: исключение IllegalArgumentException

        } catch (IllegalArgumentException e) {

            System.out.println("Длины массивов не равны");

        } catch (RuntimeException e) {

            System.out.println(e.getMessage());

        }

        try {

            int[] e = {-1, 2, 3};

            int[] f = {4, 5, 6};


System.out.println(Arrays.toString(Answer.mergeAndValidateArrays(e,
f)));  // Ожидаемый результат: исключение RuntimeException

        } catch (IllegalArgumentException e) {

            System.out.println("Длины массивов не равны");

        } catch (RuntimeException e) {

            System.out.println(e.getMessage());

        }

    }

}
```

Задача 3. Обработка исключений для разностных массивов

Реализуйте метод `subArraysWithExceptionHandling`, который принимает два массива целых чисел. Метод должен возвращать новый массив, где каждый элемент является разностью соответствующих элементов двух входных массивов. Если длины массивов не равны, выбрасывайте `IllegalArgumentException` с сообщением "Массивы разной длины". Если результат разности оказывается отрицательным, выбрасывайте `RuntimeException` с сообщением "Отрицательный результат разности".

```
import java.util.ArrayList;

import java.util.Arrays;

import java.util.List;


class Answer {

   public int[] subArraysWithExceptionHandling(int[] a, int[]
b, List<String> errors) {

       // Введите свое решение ниже


   }

}


public class Printer {

   public static void main(String[] args) {

       List<String> errors = new ArrayList<>();


       int[] a = {10, 20, 30};

       int[] b = {5, 15, 35};

       int[] result = new
Answer().subArraysWithExceptionHandling(a, b, errors);

       System.out.println(Arrays.toString(result));  //
Ожидаемый результат: исключение RuntimeException


       int[] c = {10, 20};

       int[] d = {5, 15, 25};
```

```
        result = new Answer().subArraysWithExceptionHandling(c,
d, errors);  // Ожидаемый результат: исключение
IllegalArgumentException

        System.out.println(Arrays.toString(result));



        // Проверяем, если были ошибки

        if (!errors.isEmpty()) {

            System.out.println("Обнаружены ошибки:");

            for (String error : errors) {

                System.out.println(error);

            }

        } else {

            System.out.println("Ошибок не обнаружено.");

        }

    }

}
```

Подсказка № 1

Перед тем как выполнять какие-либо операции с массивами, проверьте их длины. Используйте условие `if` для сравнения длины двух массивов. Если длины не равны, добавьте сообщение об ошибке в список `errors` и верните пустой массив.

Подсказка № 2

Создайте новый массив для хранения результатов разностей соответствующих элементов двух входных массивов. Убедитесь, что этот массив имеет длину, равную длине входных массивов.

Подсказка № 3

Пройдитесь по каждому элементу входных массивов с помощью цикла `for`. Вычислите разность соответствующих элементов и сохраните результат в новом массиве.

Подсказка № 4

После вычисления разностей, пройдитесь по новому массиву и проверьте, есть ли отрицательные значения. Если такое значение найдено, добавьте соответствующее сообщение об ошибке в список `errors`.

Эталонное решение:

```
import java.util.ArrayList;

import java.util.Arrays;

import java.util.List;



class Answer {

   public int[] subArraysWithExceptionHandling(int[] a, int[] b,
List<String> errors) {

       if (a.length != b.length) {

           errors.add("Массивы разной длины");

           return new int[0]; // Возвращаем пустой массив, так как
длины массивов не совпадают

       }



       int[] result = new int[a.length];



       for (int i = 0; i < a.length; i++) {

           result[i] = a[i] - b[i];

           if (result[i] < 0) {

               errors.add("Отрицательный результат разности в
позиции " + i + ": " + result[i]);

           }

       }
```

```
        return result;

    }

}



public class Printer {

    public static void main(String[] args) {

        List<String> errors = new ArrayList<>();



        int[] a = {10, 20, 30};

        int[] b = {5, 15, 35};

        int[] result = new Answer().subArraysWithExceptionHandling(a,
b, errors);

        System.out.println(Arrays.toString(result));  // Ожидаемый
результат: исключение RuntimeException



        int[] c = {10, 20};

        int[] d = {5, 15, 25};

        result = new Answer().subArraysWithExceptionHandling(c, d,
errors);  // Ожидаемый результат: исключение
IllegalArgumentException

        System.out.println(Arrays.toString(result));



        // Проверяем, если были ошибки

        if (!errors.isEmpty()) {

            System.out.println("Обнаружены ошибки:");

            for (String error : errors) {

                System.out.println(error);

            }
```

```
        } else {
            System.out.println("Ошибок не обнаружено.");
        }
    }
}
```

Задача 4. Поиск и замена строк

Реализуйте метод `findAndReplace`, который принимает массив строк, строку для поиска и строку для замены. Если искомая строка не найдена, добавьте сообщение об ошибке в список. Верните новый массив строк с выполненной заменой.

```
import java.util.*;
import java.util.ArrayList;
import java.util.List;


class Answer {
    public static String[] findAndReplace(String[] strings,
String target, String replacement, List<String> errors) {
        // Напишите свое решение ниже


    }
}


public class Printer {
    public static void main(String[] args) {
        List<String> errors = new ArrayList<>();
```

```
        String[] strings = {"apple", "banana", "cherry",
"date"};

        String target = "banana";

        String replacement = "orange";


        String[] result = Answer.findAndReplace(strings,
target, replacement, errors);


        System.out.println("Результаты замены: " +
Arrays.toString(result));
        if (!errors.isEmpty()) {
            System.out.println("Обнаружены ошибки:");
            for (String error : errors) {
                System.out.println(error);
            }
        }
    }
}
```

Подсказка № 1

Создайте новый массив строк, который будет содержать те же строки, что и исходный массив. Этот новый массив будет использоваться для хранения результатов замены

Подсказка № 2

Используйте цикл `for` для прохождения каждого элемента в исходном массиве строк. Для каждой строки сравните её с искомой строкой. Если строка совпадает, замените её на строку для замены в новом массиве.

Подсказка № 3

Добавьте флаг (например, `found`), чтобы отслеживать, была ли строка найдена. Если после завершения цикла флаг указывает, что строка не была найдена, добавьте соответствующее сообщение об ошибке в список ошибок.

Подсказка № 4

Проверьте каждую строку в исходном массиве и поместите либо заменённую строку, либо исходную строку в новый массив. Если строка совпадает с искомой строкой, замените её, иначе оставьте её без изменений.

Эталонное решение:

```
import java.util.*;



import java.util.ArrayList;

import java.util.List;



class Answer {

   public static String[] findAndReplace(String[] strings, String
target, String replacement, List<String> errors) {

       boolean found = false;

       String[] result = new String[strings.length];



       for (int i = 0; i < strings.length; i++) {

           if (strings[i].equals(target)) {

               result[i] = replacement;

               found = true;

           } else {

               result[i] = strings[i];

           }

       }
```

```
        if (!found) {
            errors.add("Искомая строка \"" + target + "\" не
найдена");
        }

        return result;
    }
}

public class Printer {
    public static void main(String[] args) {
        List<String> errors = new ArrayList<>();
        String[] strings = {"apple", "banana", "cherry", "date"};
        String target = "banana";
        String replacement = "orange";

        String[] result = Answer.findAndReplace(strings, target,
replacement, errors);

        System.out.println("Результаты замены: " +
Arrays.toString(result));
        if (!errors.isEmpty()) {
            System.out.println("Обнаружены ошибки:");
            for (String error : errors) {
                System.out.println(error);
            }
        }
    }
```

```
}
```