LAPORAN PRAKTIKUM
PERTEMUAN-9

Diajukan untuk memenuhi salat satu tugas praktikum Mata kuliah Pemrograman Berorientasi Objek

Disusun Oleh:
Daiva Raditya Pradipa (231511039)

Jurusan Teknik Komputer dan Informatika
Program Studi D-3 Teknik Informatika
Politeknik Negeri Bandung
2024

1. **Studi kasus 1**
   a. Screenshot Hasil Program

```
Enter a single word (letters only, please): sdasd asdasd
Not a letter
Character that make out of bonds: ' ' ot index: 5

A: 3
D: 4
S: 4
PS C:\Code\java\Pertemuan9>
```

   b. Penjelasan
      1) Run CountLetters and enter a phrase, that is, more than one word with spaces or other punctuation in between. It should throw an ArrayIndexOutOfBoundsException, because a non-letter will generate an index that is not between 0 and 25. It might be desirable to allow non-letter characters, but not count them. Of course, you could explicitly test the value of the character to see if it is between 'A' and 'Z'. However, an alternative is to go ahead and use the translated character as an index, and catch an ArrayIndexOutOfBoundsException if it occurs. Since you want don't want to do anything when a non-letter occurs, the handler will be empty. Modify this method to do this as follows:
         a. Put the body of the first for loop in a try.

```
for (int i = 0; i < word.length(); i++) {
    try {
        char currentChar = word.charAt(i);
        // if (currentChar < 'A' || currentChar > 'Z') {
        //     throw new ArrayIndexOutOfBoundsException("Not a letter \nCh
        // }
        counts[currentChar - 'A']++;
    } catch (ArrayIndexOutOfBoundsException tx) {
        System.out.println(tx.getMessage());
    }
}
```

         b. Add a catch that catches the exception, but don't do anything with it. Compile and run your program.

```
Enter a single word (letters only, please): we have dog

A: 1
D: 1
E: 2
G: 1
H: 1
O: 1
V: 1
W: 1
PS C:\Code\java\Pertemuan9>
```

         Program dapat tetap berjalan meskipun terdapat character non letter pada data yang kita inputkan pada program. Dimana sebelum kita mengimplementasikan exception pada for loop dan kita mencoba menginputkan data yang mengandung karakter non letter, program akan menampilkan error sebagai berikut

```
Enter a single word (letters only, please): asdad adasda
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: Index -33 out of bounds for length 26
        at CountLetters.main(CountLetters.java:23)
```

Hal ini dikarenakan program hanya dapat memproses data berupa kata. Maka dari itu, kita menggunakan exception di dalam for loop untuk memastikan bahwa program tetap berjalan meskipun terdapat karakter yang tidak karakter yang tidak dapat ditangan program, seperti spasi atau symbol dengan exception yang menghandle karakter-karakter tersebut. Dengan demikian, proses penghitungan huruf dalam kata akan tetap dieksekusi, meskipun ada karakter non-huruf yang muncul.

2) Now modify the body of the catch so that it prints a useful message (e.g., "Not a letter") followed by the exception. Compile and run the program. Although it's useful to print the exception for debugging, when you're trying to smoothly handle a condition that you don't consider erroneous you often don't want to. In your print statement, replace the exception with the character that created the out of bounds index. Run the program again; much nicer!

```java
public class CountLetters {
   public static void main(String[] args) {
      // Count frequency of each letter in the string
      for (int i = 0; i < word.length(); i++) {
         try {
            char currentChar = word.charAt(i);
            if (currentChar < 'A' || currentChar > 'Z') {
               throw new ArrayIndexOutOfBoundsException("Not a letter \nCharacter that make out of bonds: '" + currentChar +  "' ot index: " + i);
            }
            counts[currentChar - 'A']++;
         } catch (ArrayIndexOutOfBoundsException tx) {
            System.out.println(tx.getMessage());
         }
      }

      // Print frequencies
      System.out.println();
      for (int i = 0; i < counts.length; i++) {
         if (counts[i] != 0) {
            System.out.println((char) (i + 'A') + ": " + counts[i]);
         }
      }
   }
}
```

```
Enter a single word (letters only, please): sdasd asdasd
Not a letter
Character that make out of bonds: ' ' ot index: 5

A: 3
D: 4
S: 4
PS C:\Code\java\Pertemuan9>
```

Untuk memungkinkan karakter yang membuat program error di print atau di tampilkan pada message exception kita perlu mengimplementasikan throw exception pada code yang dapat menyebabkan error. Disini code yang dapat menyebakan error adalah saat mengcast atau convert huruf dalam kata ke dalam bentuk index atau integer.

c. Permasalahan

d. Solusi

e. Teman yang membantu

2. Studi kasus 2
   a. Screenshot hasil program

```
Enter a line of text
we have 2 dogs and 1 cats
The sum of the integers on this line is 3
PS C:\Code\java\Pertemuan9>
```

   b. Penjelasan
      1) Modify the program to add a try statement that encompasses the entire while loop. The try and opening { should go before the while, and the catch after the loop body. Catch a NumberFormatException and have an empty body for the catch.

```
try {
    while (scanLine.hasNext()) {

        val = Integer.parseInt(scanLine.next());
        sum += val;

    }
} catch (NumberFormatException e) {
}
```

      2) Compile and run the program and enter a line with mixed integers and other values. You should find that it stops summing at the first non-integer, so the line above will produce a sum of 0, and the line "1 fish 2 fish" will produce a sum of 1. This is because the entire loop is inside the try, so when an exception is thrown the loop is terminated. To make it continue, move the try and catch inside the loop. Now when an exception is thrown, the next statement is the next iteration of the loop, so the entire line is processed. The dogs-and-cats input should now give a sum of 3, as should the fish input.

```
while (scanLine.hasNext()) {
    try{
        val = Integer.parseInt(scanLine.next());
        sum += val;
    }catch(NumberFormatException e){}
}
```

3. Studi kasus 3
   a. Screenshot hasil program

```
Enter an integer: -1
Integer must be >= 0
Enter an integer: 17
The factorial method can only take values < 17
Enter an integer: 5
Factorial(5) = 120
Another factorial? (y/n)
```

b. Penjelasan

1) Modify the header of the factorial method to indicate that factorial can throw an IllegalArgumentException.

```
public static int factorial(int n) throws IllegalArgumentException {
```

2) Modify the body of factorial to check the value of the argument and, if it is negative, throw an IllegalArgumentException. Note that what you pass to throw is actually an instance of the IllegalArgumentException class, and that the constructor takes a String parameter. Use this parameter to be specific about what the problem is.

```
public static int factorial(int n) throws IllegalArgumentException {
    if (n < 0) {
        throw new IllegalArgumentException(s:"Integer must be >= 0");
    }

    int fac = 1;
    for (int i = n; i > 0; i--) {
        fac *= i;
    }
    return fac;
}
```

3) Compile and run your Factorials program after making these changes. Now when you enter a negative number an exception will be thrown, terminating the program. The program ends because the exception is not caught, so it is thrown by the main method, causing a runtime error.

```
Enter an integer: -1
Exception in thread "main" java.lang.IllegalArgumentException: Integer must be >= 0
        at kasus3.MathUtils.factorial(MathUtils.java:9)
        at kasus3.Factorials.main(Factorials.java:14)
PS C:\Code\java>
```

4) Modify the main method in your Factorials class to catch the exception thrown by factorial and print an appropriate message, but then continue with the loop. Think carefully about where you will need to put the try and catch.

```
public class Factorials {
    Run | Debug
    public static void main(String[] args) {
        String keepGoing = "Y";
        Scanner scan = new Scanner(System.in);
        while (keepGoing.equals(anObject:"Y") || keepGoing.equals(anObject:"Y")) {
            try{
                System.out.print(s:"Enter an integer: ");
                int val = scan.nextInt();
                System.out.println("Factorial(" + val + ") = " + MathUtils.factorial(val));
                System.out.print(s:"Another factorial? (y/n) ");
                keepGoing = scan.next();
            }catch(IllegalArgumentException e){}
        }
    }
}
```

5) Returning a negative number for values over 16 also is not correct. The problem is arithmetic overflow—the factorial is bigger than can be represented by an int. This can also be thought of as an IllegalArgumentException—this factorial method is only defined for arguments up to 16. Modify your code in factorial to check for an argument over 16 as well as for a negative argument. You should throw an IllegalArgumentException in either case, but pass different messages to the constructor so that the problem is clear.

```java
public static int factorial(int n) throws IllegalArgumentException {
    if (n < 0) {
        throw new IllegalArgumentException(s:"Integer must be >= 0");
    } else if (n > 16) {
        throw new IllegalArgumentException(s:"The factorial method can only take values < 17");
    }

    int fac = 1;
    for (int i = n; i > 0; i--) {
        fac *= i;
    }
    return fac;
}
```

Link github: [Tugas-PBO/Week-9 at main · RaditZX/Tugas-PBO]