Praktikum Pertemuan 9

Disusun untuk memenuhi tugas mata kuliah Pemrograman Berorientasi Objek (Praktek)

Disusun Oleh:
Dwika Ali Ramdhan (231511042)
2B – D3

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

2024

Contents

1. Case 1 : Exceptions Aren't Always Errors

1.1. Solusi 1

```
import java.util.Scanner;

public class CountLettersno1 {
    public static void main(String[] args){
        int[] counts = new int[26];
        Scanner scan = new Scanner(System.in);

         System.out.print("Masukkan suatu kata (hanya boleh huruf) : ");
         String word = scan.nextLine();

        word = word.toUpperCase();
        for (int i = 0; i < word.length(); i++) {
            try {
                counts[word.charAt(i)- 'A']++;
            } catch (ArrayIndexOutOfBoundsException e){
                return;
            }
        }

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

1.1.1. Output

```
Masukkan suatu kata (hanya boleh huruf) : DWikA12FNWI

Process finished with exit code 0
```

1.1.2. Penjelasan :

penanganan exception dengan try-catch yang kosong (silent exception handling). Ketika program menemui karakter non-huruf, exception akan ditangkap (catch) dan program langsung return (berhenti). Kelemahannya adalah program berhenti total saat menemui karakter non-huruf pertama yang ditemukan.

1.2. Solusi 2

```
import java.util.Scanner;

public class CountLettersno2 {
    public static void main(String[] args) {
        int[] counts = new int[26];
        Scanner scan = new Scanner(System.in);

        System.out.print("Masukkan suatu kata (hanya boleh huruf) : ");
        String word = scan.nextLine();

        word = word.toUpperCase();

        for (int i = 0; i < word.length(); i++) {
            try {
```

```
                counts[word.charAt(i)-'A']++;
            } catch (ArrayIndexOutOfBoundsException e) {
                System.out.println("Bukan Merupaak Huruf : " +
word.charAt(i));
            }
        }

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

1.2.1. Output

```
Masukkan suatu kata (hanya boleh huruf) : DWIka123IHFW
Bukan Merupaak Huruf : 1
Bukan Merupaak Huruf : 2
Bukan Merupaak Huruf : 3
Frekuensi Huruf:
Huruf A: 1
Huruf D: 1
Huruf F: 1
Huruf H: 1
Huruf I: 2
Huruf K: 1
Huruf W: 2
```

1.2.2. Penjelasan :

Pengembangan dari solusi pertama dengan penanganan exception yang lebih baik. Program tidak berhenti saat menemui karakter non-huruf Tetap melanjutkan perhitungan untuk semua huruf yang valid

2. Case 2 : Placing Exception Handlers

```
import java.util.Scanner;

public class ParseInts1 {
    public static void main(String[] args) {
        int val, sum = 0;
        Scanner scan = new Scanner(System.in);
        String line;
        System.out.println("\nEnter another line of text");
        Scanner scanLine = new Scanner(scan.nextLine());

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

```
        System.out.println("Version 1 - The sum of the integers on this line
is " + sum);
        scanLine.close();

        scanLine = new Scanner(scan.nextLine());
        sum = 0;

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

        System.out.println("Version 2 - The sum of the integers on this line
is " + sum);
        scanLine.close();
        scan.close();
    }
}
```

2.1.Output

```
Enter a line of text
Dwika 12 Dwika 22
Version 1 - The sum of the integers on this line is 0

Enter another line of text
Dwika 12 Dwika 22
Version 2 - The sum of the integers on this line is 34
```

2.2.Penjelasan

Terdapat 2 versi pada program ini :
Versi 1 :Try-catch block berada di luar loop while, Ketika menemui token non-integer, program akan keluar dari loop. Total penjumlahan hanya mencakup angka sebelum token non-integer pertama
Versi 2 : Try-catch block berada di dalam loop while, Setiap token non-integer akan dilewati tanpa menghentikan program, Program tetap melanjutkan pemrosesan untuk mencari angka-angka lain, Total penjumlahan bersaldari angka angka yang telah di lewati.

3. Case 3 : Throwing Exceptions

3.1.Factorials

```
import java.util.Scanner;

public class ParseInts1 {
    public static void main(String[] args) {
        int val, sum = 0;
        Scanner scan = new Scanner(System.in);
        String line;
        System.out.println("\nEnter another line of text");
        Scanner scanLine = new Scanner(scan.nextLine());

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

```
            }
        } catch (NumberFormatException e) {
            //Kosong
        }
        System.out.println("Version 1 - The sum of the integers on this line
is " + sum);
        scanLine.close();

        scanLine = new Scanner(scan.nextLine());
        sum = 0;

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

        System.out.println("Version 2 - The sum of the integers on this line
is " + sum);
        scanLine.close();
        scan.close();
    }
}
```

3.2.MathUtils

```
public class MathUtils1 {
    public static int factorial(int n) throws IllegalArgumentException {
        if (n < 0) {
            throw new IllegalArgumentException("Factorial is not defined for
negative numbers: " + n);
        }

        if (n > 16) {
            throw new IllegalArgumentException("Factorial overflow: This
method can't compute factorial for numbers greater than 16");
        }

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

3.3.Output

```
Enter an integer: 10
Factorial(10) = 3628800
Another factorial? (y/n) y
Enter an integer: 20
Error: Factorial overflow: This method can't compute factorial for numbers greater than 16
Another factorial? (y/n) y
Enter an integer: -5
Error: Factorial is not defined for negative numbers: -5
Another factorial? (y/n)
```

3.4.Penjelasan

MathUtils1:

- Menambahkan throws IllegalArgumentException pada method factorial
- Melakukan dua validasi input:
    1. Menolak angka negatif dengan exception khusus
    2. Menolak angka > 16 untuk mencegah integer overflow
- Memberikan pesan error yang spesifik untuk setiap jenis kesalahan.
-

Factorials:

- Menggunakan try-catch untuk menangkap IllegalArgumentException
- Menampilkan pesan error yang informatif ke pengguna
- Program tetap berjalan meski terjadi error
- Memungkinkan pengguna mencoba input lain
- Mencegah program crash dengan menangani exception dengan baik

LAMPIRAN

LINK GITHUB: https://github.com/DAliRIJTK/PBO_Dwika_231511042.git