Java Programming

2-4: Collections – Part 1

Practice Activities

Lesson Objectives:

- Create a collection without using generics
- Create a collection using generics
- Implement an ArrayList
- Implement a Set

Vocabulary:

Identify the vocabulary word for each definition below.

HashSet	A set similar to an ArrayList without any specific ordering.
List	An ordered Collection that may contain duplicates.
Collectiions	An interface used to define a group of objects. This includes lists and sets.
ArrayList	A list that is very similar to an array.
Set	A Collection of elements that does not contain any duplicates.

Try It/Solve It:

1. What is the difference between a set and a list?

   List = Kumpulan elemen berurutan (sequence).

   Boleh duplikasi elemen.

   Menyimpan urutan sesuai penambahan.

   Saat urutan data penting dan butuh akses berdasarkan posisi

   Set = Kumpulan elemen unik tanpa duplikasi

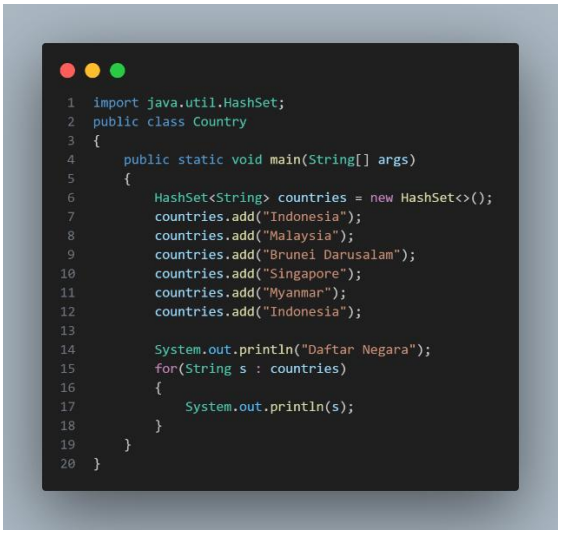
   Tidak boleh duplikasi elemen.

   Tidak menjamin urutan kecuali saat treeset

   Saat butuh data tanpa duplikasi.

2. You decide you want to roll 2 dice and see what the frequency is of each possible number combination. Would you use a Set collection to do this? State your reason(s).
   Tidak, saya tidak akan menggunakan Set collection untuk menghitung peluang kemunculan angka dari 2 dadu. Karena peluang angka yang muncul bisa berulang jika menggunakan set makan hal berulang tersebut tidak akan masuk dikategorikan duplikat. Untuk menghitung peluang dari 2 dadu ini lebih baik menggunakan List karena mengizinkan duplicate data.

3. Using a collection create a variable that will store a list of countries (Strings). Your collection should not store duplicates, and order is not important. Test your code by adding 6 countries, one of which is a duplicate.



```java
import java.util.HashSet;
public class Country
{
    public static void main(String[] args)
    {
        HashSet<String> countries = new HashSet<>();
        countries.add("Indonesia");
        countries.add("Malaysia");
        countries.add("Brunei Darusalam");
        countries.add("Singapore");
        countries.add("Myanmar");
        countries.add("Indonesia");

        System.out.println("Daftar Negara");
        for(String s : countries)
        {
            System.out.println(s);
        }
    }
}
```

```
Daftar Negara
Myanmar
Singapore
Malaysia
Brunei Darusalam
Indonesia
```

   HashSet dipilih karena tidak menyimpan elemen duplikat dan tidak mempertahankan urutan.
   Negara “Indonesia” hanya akan muncul satu kali meskipun ditambahkan dua kali.

4. Would the following Collection.sort() statements both work? Explain your answer.

```
HashSet<String> countriesSet = new HashSet<String>(); Collections.sort(countriesSet);
ArrayList<String> countriesList = new ArrayList(); Collections.sort(countriesList);
```

   Collections.sort() hanya dapat menerima objek yang mengimplementasikan interface List. Karenanya hanya Collections.sort(countriesList) yang berhasil karena bertipe ArrayList yang merupakan subclass dari interface List. Jika countriesset tidak berhasil karena subclass interface Set.

5. Below is a user implementation of a Stack using arrays.
   - push adds an item to the Stack
   - pop removes an item from the stack
   - isEmpty return a Boolean value of true if the Stack is empty

   Convert this to a generic implementation using an ArrayList.

```java
public class ArrayStack {
      private int maxsize;
      private int top;
      private int[] items;
```

```
public ArrayStack(int maxsize) {
      if (maxsize <= 0)
            throw new ArrayStackException(
                        "Stack size must be positive");
      items = new int[maxsize];
      this.maxsize = maxsize;
      top = 0;
}

public void push(int item) {
      if (top == items.length)
            throw new ArrayStackException("Overflow Error");
      items[top] = item;
      top++;
}


public int pop() {
      if (isEmpty())
            throw new ArrayStackException("Underflow Error");
      return items[--top];
}

public boolean isEmpty() {
      return (top == 0);
}

public static class ArrayStackException extends RuntimeException {
      public ArrayStackException(String message) {
            super(message);
      }
}


public static void main(String[] args) {
       ArrayStack stack = new ArrayStack(3);
       stack.push(1);
       stack.push(2);
       stack.push(3);
       //stack.push(4);  //overflow error
       System.out.println(stack.pop());
       System.out.println(stack.pop());
       System.out.println(stack.pop());

       }
}
```

```java
import java.util.ArrayList;

public class ArrayStack<T> {
    private ArrayList<T> items;

    public ArrayStack() {
        items = new ArrayList<>();
    }

    // Menambahkan elemen ke atas stack
    public void push(T item) {
        items.add(item);
    }

    // Menghapus dan mengembalikan elemen teratas
    public T pop() {
        if (isEmpty()) {
            throw new RuntimeException("Underflow Error: Stack kosong");
        }
        return items.remove(items.size() - 1);
    }

    // Mengecek apakah stack kosong
    public boolean isEmpty() {
        return items.isEmpty();
    }

    // Program utama untuk menguji stack
    public static void main(String[] args) {
        ArrayStack<Integer> stack = new ArrayStack<>();
        stack.push(1000);
        stack.push(2000);
        stack.push(3000);

        System.out.println(stack.pop()); // 3
        System.out.println(stack.pop()); // 2
        System.out.println(stack.pop()); // 1
    }
}

```