第 3 回 Java 講座

今回の内容

今週の Java 講座はコレクション、拡張 for 文, ガベージコレクションについて扱う.今週の Java 講座は一番内容が薄いものになるだろう.

コレクション

コレクションとは大きさが決まっていない配列だと考えればよい.コレクションには

- List　先頭の要素要素から最後までが直線的に直結している構造
- Set　同じものは含まないという構造.要素間につながりはない
- Map　各要素がキーを持ち、キー値により検索できる.要素間につながりはない
- Queue　スタックやキューによる出し入れを提供する.

の 4 タイプがある.今回は List について扱う.MAP については演習問題を参照せよ.

List

List は先頭の要素要素から最後までが直線的に直結している構造である. データ構造とアルゴリズムを履修している人は、リストについて学習していると思われるが、Java では標準でリスト構造が準備されている.List には Array List, Linked List, Vector の 3 つに分かれる.

Array List は要素サイズが動的変化する配列のようなものである. Linked List はキューやスタックの機能を持つ双方向のリストである.　Vector はマルチスレッド環境で安全に動作するように設計されたリストである.

次のサンプルソースを実行してみよう!!

```java
/*
 * Sample1.java
 * ArrayListの使い方
 * 普通のfor文を使用
 */
import java.util.ArrayList;

public class Sample1 {
	public static void main(String[] args) {
		// コレクションオブジェクト(ArrayList型)の生成
		// ※コレクションが扱うデータ型としてDataSampleを指定する例
		ArrayList<String> list = new ArrayList<>();

		System.out.println("Section1:データの挿入を行う");
		// データを挿入する
		// Data型の適当なオブジェクトを作成して挿入を行う
		list.add("長門");
		list.add("陸奥");
		list.add("伊勢");
		list.add("日向");
		list.add("雪風");
		list.add("赤城");
		list.add("加賀");

		// 要素数の確認方法
		//size()で行うことが可能
		System.out.println("要素数は"+list.size()+"です.");

		System.out.println("Section2:データの表示をfor文で行う");
		// データの表示をfor文で行う
		// ArrayListの中身を１つずつ取り出して表示してみる
		for(int i=0; i<list.size(); i++){
			String Ship = list.get(i);
			System.out.println(Ship);
		}

	}
}
```

Map 型

HashMap は連想配列とも呼ばれ、要素を格納する時に要素に対応するキーを合わせて登録する.例えば、商品コードをキーとして商品名を登録すれば、商品コードを使って商品名を知ることが出来る.

拡張 for 文

連続したデータ型(配列やコレクションなど)の要素すべてに対して同じ処理を行うときに利用する.for 文より便利に使用することが可能である.ほかの言語(例えば php など)に触れたことがある人は、for-each 文と表現した方が分かりやすいかも知れない.

```
/*
 * Sample2.java
 * ArrayListの使い方
 * 拡張for文(for-each)の使い方
 */

import java.util.ArrayList;

public class Sample2 {

	public static void main(String[] args) {
		ArrayList<String> list = new ArrayList<>();

		System.out.println("Section1:データの挿入を行う");
		// データを挿入する
		// Data型の適当なオブジェクトを作成して挿入を行う
		list.add("長門");
		list.add("陸奥");
		list.add("伊勢");
		list.add("日向");
		list.add("雪風");
		list.add("赤城");
		list.add("加賀");

		System.out.println("Section2:データの表示をfo-eachr文で行う");
		// データの表示をfor-each文で行う
		// ArrayListの中身を１つずつ取り出して表示してみる
		for(String Ship : list){
			System.out.println(Ship);
		}
	}

}
```

ガベージコレクション

ガベージコレクション(GC)とはプログラムが動的確保したメモリ領域のうち、使わなくなったメモリ領域を解放することである.
C 言語では malloc 関数などで動的に確保したメモリ領域を free 関数で解放する必要があったが、Java では JVM が自動で行ってくれる.

演習問題

良い演習問題が考えられなかったため、次のサンプルソースを入力して、プログラムの動きを確認しなさい.

(1) Map を利用した場合の例について、Sample3.java をプログラムして確認しなさい.

```
/*
 * Sample3.java
 * Mapの使い方
 */

import java.util.HashMap;
import java.util.Map;
import java.util.Set;

public class Sample3 {

	public static void main(String[] args) {
		// 以下の1つだけコメントを外して、実行結果を比べる
		// 順番通りになっていない(保存した通りに表示している)
		Map<String, String> apostle = new HashMap<>();
		//入力された順番通りに出力
		//Map<String, String> apostle = new LinkedHashMap<>();
		//1文字1文字比較して(asciiコード表通り)表示する
		//Map<String, String> apostle = new TreeMap<>();


		// 適当に使徒の名前を入力
		//第一引数=キー, 第二引数=値
		apostle.put("第1使徒","アダム");
		apostle.put("第2使徒","リリス");
		apostle.put("第5使徒","ラミエル");
		apostle.put("第4使徒","シャムシエル");
		apostle.put("第3使徒","サキエル");
		apostle.put("第18使徒","リリン");
		apostle.put("第17使徒","タブリス");

//次のページへ続く
```

```
                // Setを使ってキーを取り出す.
                Set<String> keys = apostle.keySet();

                // 取り出したキーを表示させる(for-each文)
                for(String str : keys){
                        System.out.println(str +"-"+ apostle.get(str) + "¥t");
                }
        }

}
```

(2) Array List を利用した場合の例について Sample4.java と SampleData.java をプログラミングし確認しなさい.

```
/*
 * Sample4.java
 * ArrayListの補足
 * データの挿入と削除
 */
import java.util.ArrayList;

public class Sample4 {

        public static void main(String[] args) {
                // コレクションオブジェクト(ArrayList型)の生成
                // ※コレクションが扱うデータ型としてSampleDataを指定する場合
                ArrayList<SampleData> shiratuyu = new ArrayList<>();


                System.out.println("section1: データの挿入");
                // section1: データの挿入
                // SampleData型のオブジェクトを作成してデータを追加する.
                shiratuyu.add(new SampleData("白露"));
                shiratuyu.add(new SampleData("時雨"));
                shiratuyu.add(new SampleData("村雨"));
                shiratuyu.add(new SampleData("夕立"));
                shiratuyu.add(new SampleData("春雨"));
                shiratuyu.add(new SampleData("涼風"));
                // 要素数を確認する
                System.out.println("要素数は"+shiratuyu.size()+"です.");

                //改行して終了
                System.out.println("");


//次のページへ続く
```

```
                System.out.println("section2: データの表示");
                // section2: データの取り出し(for-each文)
                for(SampleData distroyer : shiratuyu){
                        System.out.println("白露型 : "+distroyer.getVal());
                }

                //改行して終了
                System.out.println("");

                System.out.println("secction3: データの個別表示");
                // secction3: データの取り出し(要素番号を指定)
                // 要素番号は0から開始
                // 3番艦の村雨を表示
                SampleData ship = shiratuyu.get(2);
                System.out.println("白露型(2): "+ship.getVal());
                //改行して終了
                System.out.println("");

                System.out.println("section4: データの途中挿入");
                // リスト構造の利点が分かる
                // section4: 要素の挿入
                // 4番目と5番目の間に新しい要素を挿入する
                shiratuyu.add(5,new SampleData("五月雨"));
                // 要素数の確認
                System.out.println("現在の要素数は"+shiratuyu.size()+"です.");
                // 全件を表示して確認
                for(SampleData distroyer : shiratuyu){
                        System.out.println("白露型 : "+distroyer.getVal());
                }
                //改行して終了
                System.out.println("");

                System.out.println("section5:データの削除");
                // section5: 要素の削除
                // 4番目の要素を取り除く(春雨はアーケード版で実装されていないため)
                shiratuyu.remove(4);
                // 要素数の確認
                System.out.println("現在の要素数は"+shiratuyu.size()+"です.");
                // 全件を表示して確認
                for(SampleData distroyer : shiratuyu){
                        System.out.println("白露型 : "+distroyer.getVal());
                }
        }

}
```

```java
/*
 * SampleData.java
 * Sample4.javaで使用するクラス
 */

public class SampleData {
    String value;  // 適当な文字列を入れるStringオブジェクト

    // 引数なしのコンストラクタ
    public SampleData(){
        value = "";
    }

    // 引数ありのコンストラクタ
    public SampleData(String s){
        this.value = s;
    }

    // valueに引数で与えられた文字列を代入
    public void setVal(String s){
        this.value = s;
    }

    // 現在のvalueの値を返す
    // ゲッター
    public String getVal(){
        return value;
    }

}
```