2023 年 1 月 25 日（木）実施

応用プログラム（3） ― キー検索 ―

コレクションには，**ハッシュテーブル**と呼ばれるものがある。これは，**キー(key)と値(value)と**を組として保持しているものである。通常の配列が添字により各要素にアクセス出来るのに比べて，ハッシュテーブルではキーを用いて各値にアクセスすることが出来る。キー及びそのキーから連想される値の組を保持していることから，ハッシュテーブルは**連想配列**とも呼ばれる。

ハッシュテーブルの特長は，指定したキーから対応する値を高速に得られることであり，ここでは，この機能を利用して**キー検索**を行うプログラムを扱う。

Java 言語ではハッシュテーブルは，**HashMap クラス**を用いて実現される。HashMap クラスは，**Map インタフェース**と呼ばれる，キーと値との**マッピング**(関連付け)を表すデータ構造のためのインタフェースの，ハッシュテーブルに基づく実装である。

HashMap クラスには引数無しのコンストラクタが定義されていて，次の様にマップとしてインスタンスを生成する。(初期容量を指定するコンストラクタや，初期容量及び負荷係数を指定するコンストラクタもある)

```
Map<キーのデータ型,値のデータ型> 変数名 = new HashMap<キーのデータ型,値のデータ型>();
```

例) `Map<String, Integer> hmi = new HashMap<String, Integer>();`

Search クラス

Eclipse で jimbo の様に『自分の名前』のパッケージを指定して，次の Search クラスを作成せよ。(これは直接実行出来ないクラス ⇒ 後の例題及び演習に利用する)

```
package jimbo;

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

public class Search {
    Map<String, Integer> hmi = new HashMap<String, Integer>();
    Map<String, String> hms = new HashMap<String, String>();

    Search(String[] str1, int[] num){
        InitHMi(str1, num);
    }

    Search(String[] str1, String[] str2) {
        InitHMs(str1, str2);
    }

    void InitHMi(String[] istr1, int[] inum) {
        for (int i=0; i<istr1.length; i++)
            hmi.put(istr1[i], inum[i]);
    }
```

```
    void InitHMs(String[] istr1, String[] istr2) {
        for (int i=0; i<istr1.length; i++)
            hms.put(istr1[i], istr2[i]);
    }

    void getSeti() {
        for (String str : hmi.keySet())
            System.out.println(str + ": " + hmi.get(str));
    }

    void getSets() {
        for (String str : hms.keySet())
            System.out.println(str + ": " + hms.get(str));
    }

    void geti(String str) {
        System.out.println(str + ": " + hmi.get(str));
    }

    void gets(String str) {
        System.out.println(str + ": " + hms.get(str));
    }
}
```

【解説】

1. Search クラスでは，コンストラクタから InitHMi メソッドまたは InitHMs メソッドを呼び出してハッシュテーブル hmi または hms の初期化を行う。
2. コンストラクタとしては，文字列及び整数の配列を引数として受け取るものと，2 つの文字列の配列を引数として受け取るものとの 2 種類が用意されていて，その引数をそれぞれ InitHMi メソッドまたは InitHMs メソッドに受け渡し，HashMap クラスの put メソッドによってハッシュテーブルのマップに関連付けていく。
3. put メソッドは，指定されたキーと指定された値とをマップに関連付ける。
4. keySet メソッドは，マップに含まれるキーと値との組を返す。
5. for (String str : hmi.keySet())は，**拡張 for ループ**と呼ばれ，for（宣言 : 参照）文 の形でコレクションの要素を順に参照することが出来るものである。ここで，『宣言』には，配列の要素の型またはコレクションの要素の型，及び任意の名前を指定し，『参照』には，配列名またはコレクションの名前を指定する。
6. get メソッドは，指定されたキーがマップされている値を返し，そのキーのマッピングがこのマップに含まれていない場合は null を返す。

例題 1

次のプログラムは，国名をキーとし，番号を値とするハッシュテーブルを作成し，キー検索を行うものである。これを入力し，ビルドして，実行せよ。ここで，クラス名は Sample13_1，ソースファイル名は Sample13_1.java とする。

```
package jimbo;

public class Sample13_1 {

    public static void main(String[] args) {
        // TODO 自動生成されたメソッド・スタブ

        // キー
        String[] country = {"日本", "中国", "ネパール", "インド", "ベトナム"};
        // 値
        int[] no ={100, 50, 95, 90, 80};

        Search search = new Search(country, no);

        System.out.println("----------------------");
        System.out.println("ハッシュテーブルの表示");
        System.out.println("----------------------");
        search.getSeti();

        System.out.println();
        System.out.println("----------------------");
        System.out.println("キー検索の結果");
        System.out.println("----------------------");
        search.geti(country[2]);
    }

}
```

例題 2

次のプログラムは，名前の読みをキーとし，名前を値とするハッシュテーブルを作成し，キー検索を行うものである。これを入力し，ビルドして，実行せよ。ここで，クラス名は Sample13_2，ソースファイル名は Sample13_2.java とする。

```
package jimbo;

import java.util.Scanner;

public class Sample13_2 {

    public static void main(String[] args) {
        // TODO 自動生成されたメソッド・スタブ
        String[] romaji, friend;
        final int NUM = 5;
        romaji = new String[NUM];
        friend = new String[NUM];
        String str1 = null, str2 = null, str3 = null;
        Scanner sc = new Scanner(System.in);

        for (int i=0; i<NUM; i++) {
            System.out.print((i+1)+"人目の友人の名前(姓)を入力してください：");
            str2 = sc.next();
            System.out.print("名前の読みをローマ字で入力してください：");
```

```
            str1 = sc.next();
            romaji[i] = str1;
            friend[i] = str2;
        }

        Search search = new Search(romaji, friend);

        System.out.println();
        System.out.println("----------------------");
        System.out.println("ハッシュテーブルの確認");
        System.out.println("----------------------");
        search.getSets();

        System.out.println();
        System.out.println("-----------");
        System.out.println("キー検索");
        System.out.println("-----------");

        System.out.print("検索したい名前の読みをローマ字で入力してください：");
        str3 = sc.next();
        search.gets(str3);
    }

}
```

演習

次のプログラムリストの空欄を埋めたプログラムを作成し，ビルドして実行せよ。ここで，クラス名は Ex13，ソースプログラム名は Ex13.java とする。（Eclipse では，workspace の直下の Prog1 に，ダウンロードした product.csv を配置してから実行する）

```
package jimbo;

import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.util.Scanner;

public class Ex13 {

    public static void main(String[] args) {
        // TODO 自動生成されたメソッド・スタブ
        String[] code, prod;
        Scanner 1)      = new Scanner(System.in);
        String 2)       = "product.csv";
        final int readAheadLimit = 1024*1024;
        int count = 0;
        String strc = null;

        try {
            BufferedReader br = new BufferedReader(new FileReader(fname));
```

```
            br.mark(readAheadLimit);

            String  3)      = null;

            while ((iline = br.readLine()) != null) {
                count++;
            }

            br.reset();

            code = new String[count];
            prod = new String[count];

            for (int i = 0; (iline = br.readLine()) != null; i++) {
                String[] str =iline.split(",");
                code[i] = str[0];
                prod[i] = str[1];
            }

            br.close();

            Search search = new Search(code, prod);

            System.out.println("----------------------");
            System.out.println("ハッシュテーブルの確認");
            System.out.println("----------------------");
            search.4)      ();

            System.out.println();
            System.out.println("-----------");
            System.out.println("キー検索");
            System.out.println("-----------");

            System.out.print("検索したい製品のコードを入力してください：");
            strc = sc.next();
            search.5)      (strc);
        } catch (FileNotFoundException fe) {
            System.out.println(fname +"というファイルが開けません");
        } catch (IOException err) {
            System.out.println("IOException をキャッチ"+err);
        }
    }

}
```

【解説】

1. mark メソッドは，ストリームの現在位置にマークを設定する。ここでは，ファイルの先頭位置を記録しておく。
2. while 文では，ファイルの行数を数えている。
3. reset メソッドは，直前のマーク位置にストリームをリセットする。

提出物：

1）例題 1，例題 2 及び演習のプログラムのコンソールへの出力結果をコピーして貼り付けたテキストファイル res13. txt をメールに添付する。

2）演習で空欄に適切な語句を埋めたソースプログラムのファイル Ex13.java をメールに添付する。

3）第 13 回理解度確認用の質問ファイル Prog1_Questions_13th.txt に解答を記入して，メールに添付する。

＊ メールの件名は『プログラミング 1　第 13 回課題』（鍵括弧は要らない）とする。