独習Java

10.4 VectorクラスとEnumerationインターフェイス

10.5 Stackクラス

10.6 Hashtableクラス

10.7 StringTokenizerクラス

発表者

佐々木研　　田島勇樹

１０．４ Vectorクラス（1）

- Vectorクラス

→要素を追加した分だけ自動的にサイズが拡張される一種の動的配列を作成
最初に初期サイズを決めて作成する。

- Vectorコンストラクタ

Vector()　　・・・１０個の要素を格納できるベクトルが作成される

Vector(int n)　　・・・n個の要素を格納できるベクトルが作成される

Vector(int n,int delta)　・・・初期サイズがn個の要素で、delta個の要素を単位として拡張される

１０．４　Vectorクラス（2）

- Vectorクラスに定義されている主なインスタンスメソッド（1）

メソッド	説明
void addElement(Object obj)	objをベクトルに追加する
int capacity()	ベクトルのサイズを返す
Object clone()	現在のオブジェクトの複製を返す
boolean contains(Object obj)	objが現在のベクトルに含まれている場合に真を返す それ以外の場合は、偽を返す
void copyInto(Object array[])	現在のオブジェクトの要素をarrayにコピーする
Object elementAt(int index)	indexで指定された位置の要素を返す
Enumeration elements()	要素の列挙を返す
void ensureCapacity(int minimum)	最小サイズをminimumに設定する
Object firstElement()	１つ目の要素を返す
int indexOf(Object obj)	ベクトル内でobjを検索し、最初に見つかったobjの位置を返す。objがなかった場合は-1を返す。
int indexOf(Object objn,int index)	ベクトル内でindexで指定された位置からobjを検索し、最初に見つかったobjの位置を返す。Objがなかった場合、-1を返す
void insertElementAt(Object obj,int index)	objをindexで指定された位置に追加する

１０．４ Vectorクラス (3)

- Vectorクラスに定義されている主なインスタンスメソッド (2)

メソッド	説明
boolean isEmpty()	ベクトルが空の場合真を返す。それ以外の場合は偽を返す。
Object lastElement()	最後の要素を返す。
int lastIndexOf(Object obj)	ベクトル内でobjを検索し、最後に見つかったobjの位置を返す。objがなかった場合は-1を返す。
int lastIndexOf(Object obj,int index)	ベクトル内でindexで指定された位置からobjを検索し、最後に見つかったobjの位置を返す。objがなかった場合は-1を返す。
void removeAllElements()	全ての要素を削除する。
boolean removeElement(Object obj)	ベクトルの最初のobjを削除する。削除できた場合に真を返し、それ以外の場合に偽を返す。
void removeElementAt(int index)	Indexで指定された位置の要素を削除する。
void setElementAt(Object obj,int index)	Indexで指定された位置の要素にobjを代入する。
void setSize(int size)	要素の数をsizeに変更する。sizeからはみ出した要素は破棄される。
int size()	ベクトルに格納されている要素の数を返す。
String toString()	ベクトルを文字列で表す。
void trimToSize()	ベクトルのサイズを要素の数で切り詰める。

１０．４Enumerationインターフェイス

- Enumerationインターフェイス

→オブジェクト群に対して一定の処理を繰り返すために使用

Enumerationインターフェイスで定義されているメソッド

hasMoreElements()メソッド
boolean hasMoreElements()

nextElement()メソッド
Object nextElement()

hasMoreElements()メソッドはまだ要素がある場合に真を返す。それ以外は偽を返す。
nextElement()メソッドは次にある要素を返す。

１０．４　VectorクラスとEnumerationインターフェイス（1）

VectorクラスとEnumerationインターフェイスを使用したプログラム

```
import java.util.*;
class EnumerationDemo{
public static void main(String args[]){

//ベクトルとその要素を作成
Vector vector =new Vector();
vector.addElement(new Integer(5));
vector.addElement(new Float(-14.14f));
vector.addElement(new String(“Hello”));
//ベクトルの要素を表示する
Enumeration e=vector.elements();
while(e.hasMoreElements()){
Object obj=e.nextElement();
System.out.println(obj);
}
}
}
```

出力結果

```
5
-14.14f
Hello
```

１０．４　VectorクラスとEnumerationインターフェイス（2）

- コンパイルする上での注意点

前のプログラムをそのままコンパイルすると警告が出る。
警告「VectorDemo.javaの操作は未チェックまたは安全ではありません」

→Vectorというオブジェクトが全ての型を許容するため

-Xlintオプションをつけてコンパイルするとオブジェクトを操作している箇所についての一連の警告が表示

Objectクラスに由来するオブジェクト全般を扱うため<Object>をつける。
Vector<Object> vector=new Vector<Object>();

１０．５ Stackクラス（1）

- Stackクラス

→Vectorクラスを拡張し、LIFO型(最後に入れたものを最初に取り出す方式)スタックを提供

メソッド	説明
boolean empty()	スタックが空の場合に真を返す。それ以外の場合は偽を返す。
Object peek() throws EmptyStackException	スタックの一番上のオブジェクトを返す（ただし、オブジェクトは削除しない）
Object pop() throws EmptyStackException	スタックの一番上のオブジェクトを返し、それをスタックから削除する。
Object push(Object obj)	objをスタックにプッシュし、objを返す
int search(Object obj)	スタック内でobjを検索する。objが見つからない場合は-1を返す。見つかった場合は、objの位置を示すインデックスを返す。スタックの一番上はインデックス1

１０．５　Stackクラス（２）

- Stackクラスを使用したプログラム

```
import java.util

class PushPop{
public static void main(String args[]){

//スタックを作成
Stack stack=new Stack();

//要素をスタックにプッシュ
for(int i=0;i<args.length;i++)
stack.push(new Integer(args[i]));

//スタックから要素をポップして表示
while(!stack.empty()){
Object obj=stack.pop();
System.out.println(obj);
}
}
}
```

入力と出力

```
>java PushPoP 1 2 3 4 5
5
4
3
2
1
```

１０．６ Hashtableクラス（1）

- Hashtableクラス

ハッシュ表の各エントリにはキーと値が含まれる。
キーと値：オブジェクト（キーは一意、値は一意でなくてよい）
値を取り出す場合はキーを使う→連想配列

- Hashtableコンストラクタ

Hashtable()
Hashtable(int n)
Hashtable(int n,float lf)

n:ハッシュ表の初期サイズ
lf:荷重率(load factor:有効なデータが全体のサイズに占める割合）
lfに指定できる値は0~1
エントリ数がサイズと荷重率の積を超えた場合
→ハッシュ表のサイズが自動的に拡張

10．6 Hashtableクラス（2）

- Hashtableクラスに定義されている主なインスタンスメソッド

メソッド	説明
void clear()	このハッシュ表を空にする。
boolean contains(Object v) throws NullPointerException	ハッシュ表の値としてvが含まれている場合に真を返す。それ以外の場合に偽を返す。vがnullの場合にNullPointerExceptionを投げる。
boolean containsKey(Object k)	ハッシュ表のキーとしてkが含まれている場合に真を返す。それ以外の場合に偽を返す。
boolean containsValue(Object v)	ハッシュ表の値としてvが含まれている場合に真を返す。それ以外の場合に偽を返す。
Enumeration elements()	値の列挙を返す。
Object get(Object k)	キーkに関連付けられた値を返す。
boolean isEmpty()	ハッシュ表が空の場合に真を返す。それ以外の場合に偽を返す。
Enumeration keys()	キーの列挙を返す。
Object put(Object k,Object v) throws NullPointerException	1つのキー/値ペアをハッシュ表に追加する（kがキーで、vが値）kまたはvがnullの場合にNullPointerExceptionを投げる
Object remove(Object k)	Kをキーに持つキー/値ペアを削除する
int size()	キーの数を返す
String toString()	このハッシュ表を表す文字列を返す

１０．６ Hashtableクラス (3)

- Hashtableクラスを使った例

```
import java.util.*;
class HashtanleDemo{
public static void main(String args[]){

//ハッシュ表を作成し、情報を設定する
Hashtable hashtable=new Hashtable();
hashtable.put(“apple”,”red”);
hashtable.put(“strawberry”,”red”);
hashtable.put(“lime”,”green”);
hashtable.put(“banana”,”yellow”);
hashtable.put(“orange”,”orange”);
//ハッシュ表の要素を表示
Enumeration e=hashtable.keys();
while(e.hasMoreElements()){
Object k=e.nextElement();
Object v=hashtable.get(k);
System.out.println(“key=”+k+
”;value=“+v);
}
//appleの値を表示
System.out.print(“\nThe color of
    an apple is: “);
Object v=hashtable.get(“apple”);
System.out.println(v);
}
}
```

１０．６　Hashtableクラス (4)

- Hashtableクラスを使ったプログラムの出力結果

key=banana;value=yellow
key=apple;value=red
key=orange;value=orange
key=lime;value=green
key=strawberry;value=red

The color of an apple is:red

注意：ハッシュ表に追加した順番と取り出したときの順番が必ずしも一致はしない。
　　どのように格納されるかはJVMの実装によって異なる。

同様にコンパイルをすると「ジェネリックス」という仕組みにより警告が出る。
Hashtable<String,String>hashtable=new Hashtable<String,String>();
上記のように記述すると警告は表示されない。

１０．７　StringTokenizerクラス（1）

- StringTokenizerクラス

→文字列をトークン（文字列の構成要素）に分解するために使う
トークンの区切り文字として使われる文字を指定することが可能

例　ファイルに含まれているデータをフォーマットするプログラム

StringTokenizerコンストラクタ
StringTokenizer(String str)
StringTokenizer(String str,String delimiters)
StringTokenizer(String str,String delimiters,boolean delimitersAreTokens)

str:分解対象となる文字列
delimiters:トークンの区切り文字
delimitersAreTokensが真：区切り文字がトークンとして返される
delimitersAreTokensが偽：区切り文字は返されない

１０．７　StringTokenizerクラス（2）

StringTokenizerクラスはEnumerationインターフェイスが実装される

StringTokenizerクラスに定義されている主なインスタンスメソッド

メソッド	説明
int countTokens()	文字列に含まれているトークンの数を返す。
boolean hasMoreTokens()	まだトークンがある場合に真を返す。それ以外の場合に偽を返す。
String nextToken()	次のトークンを返す。
String nextToken(String delimiters)	delimitersを区切り文字群として設定し、次のトークンを返す。

１０．７　StringTokenizerクラス (3)

- StringTokenizerクラスを使ったプログラム

```
import java.util.*;
class StringTokenizerDemo{
 public static void main(String args[]){
  String str=“123/45.6/-11.2/100/-20”;
  StringTokenizer st=new StringTokenizer(str,”/”);
  while(st.hasMoreTokens()){
   String s=st.nextToken();
   System.out.println(s);
  }
 }
}
```

出力結果

123

45.6

-11.2

100

-20

練習問題1

1．ベクトルを使ってA～Zを格納し、文字と文字の位置を検索するプログラムを作成せよ。

2．Stackクラスを用いて逆ポーランド電卓を作成せよ。

例　>java 10 20 + 3 /

10.0

>java 1 2 3 + +

6

練習問題2

3. 都市の名前をコマンドライン引数として受け取り、その都市がある国の名前を表示するプログラムを作成せよ。都市名と国名はハッシュ表に保存する。コマンドライン引数が指定されなかったり、ハッシュ表にない都市名が指定された場合エラーメッセージを表示せよ。