**package** stack;

**import** java.util.\*;

**public** **class** ArrayListStack<T> **implements** IStack<T>{

//--------------------------------------------------

// Body of the stack is held in an ArrayList

**private** ArrayList<T> stackBody;

//--------------------------------------------------

/\*\*

\* Constructor: Instantiates the body

\*/

**public** ArrayListStack()

{

stackBody = **new** ArrayList<T>();

}

//--------------------------------------------------

/\*\*

\* Test for emptiness: size == 0 means stack is empty

\*/

**public** **boolean** isEmpty()

{

**return** (stackBody.size() == 0);

}

//--------------------------------------------------

/\*\*

\* Insert a new item at the top of the stack

\*/

**public** **void** push(T item)

{

stackBody.add(item);

}

//--------------------------------------------------

/\*\*

\* Delete the top-most element of the stack

\* Report error and return null if stack is empty

\*/

**public** T pop()

{

**if** (isEmpty())

{

System.***out***.println("Error in ArrayStack.pop() Stack Empty ");

**return** **null**;

}

**else**

{

T topElement = stackBody.get(stackBody.size()-1);

stackBody.remove(stackBody.size()-1);

**return** topElement;

}

}

//--------------------------------------------------

/\*\*

\* Fetch the top-most element of the stack without

\* removing it. Report error and return null if stack

\* is empty

\*/

**public** T top()

{

**if** (isEmpty())

{

System.***out***.println("Error in ArrayStack.top() Stack Empty ");

**return** **null**;

}

**else**

{

T topElement = stackBody.get(stackBody.size()-1);

**return** topElement;

}

}

**boolean** x;

**public** **boolean** equals(ArrayListStack<T> otherStack)

{

**for**(**int** k =0; k<stackBody.size(); k++)

**if** (otherStack.stackBody.get(k) != stackBody.get(k))

x = **false**;

**else**

x = **true**;

**return** x;

}

**public** **int** size()

{

**return** (stackBody.size());

}

}

**package** queue;

**import** java.util.\*;

**import** stack.ArrayListStack;

**public** **class** ArrayListQueue<T> **implements** IQueue<T> {

ArrayList<T> body;

**public** ArrayListQueue()

{

body = **new** ArrayList<T>();

}

**public** **boolean** isEmpty()

{

**return** (body.size()== 0);

}

**public** **void** insert(T item)

{

body.add(item);

}

**public** T delete()

{

**if** (isEmpty())

**return** **null**;

T item = body.get(0);

body.remove(0);

**return** item;

}

**public** T front()

{

**if** (isEmpty())

**return** **null**;

T item = body.get(0);

**return** item;

}

**public** **void** printQueue()

{

**for**(**int** i = 0; i < body.size(); i++)

{

System.***out***.print(body.get(i) + " ");

}

System.***out***.println();

}

**boolean** x;

**public** **boolean** equals(ArrayListQueue<T> otherQueue)

{

**for**(**int** k =0; k<body.size(); k++)

**if** (otherQueue.body.get(k) != body.get(k))

x = **false**;

**else**

x = **true**;

**return** x;

}

**public** **int** size()

{

**return** body.size();

}

}

**package** queue;

**import** stack.ArrayListStack;

**public** **class** Tester{

**public** **static** **void** main(String [] args)

{

ArrayListStack<String> s1 = **new** ArrayListStack<String>();

ArrayListStack<String> s2 = **new** ArrayListStack<String>();

ArrayListStack<String> s3 = **new** ArrayListStack<String>();

ArrayListQueue<String> q1 = **new** ArrayListQueue<String>();

ArrayListQueue<String> q2 = **new** ArrayListQueue<String>();

ArrayListQueue<String> q3 = **new** ArrayListQueue<String>();

String dataForS1 = "Not everything can be counted counts,"

+ " and not everything that counts can be counted";

String dataForS2 = "Not everything can be counted counts,"

+ " and not everything that counts can be counted";

String dataForS3 = "Only two things are infinite, the universe and human stupidity, and I'm not sure about the former";

String dataForQ1 = "Not everything can be counted counts,"

+ " and not everything that counts can be counted";

String dataForQ2 = "Not everything can be counted counts,"

+ " and not everything that counts can be counted";

String dataForQ3 = "Only two things are infinite, the universe and human stupidity, and I'm not sure about the former";

String [] strings0 = dataForS1.split("\\s+");

**for**(**int** k = 0; k<strings0.length;k++)

{s1.push(strings0[k]);}

String [] strings1 = dataForS2.split("\\s+");

**for**(**int** k = 0; k<strings1.length; k++)

s2.push(strings1[k]);

String [] strings2 = dataForS3.split("\\s+");

**for**(**int** k = 0; k<strings2.length;k++)

s3.push(strings2[k]);

String [] strings3 = dataForQ1.split("\\s+");

**for**(**int** k = 0; k<strings3.length;k++)

q1.insert(strings3[k]);

String [] strings4 = dataForQ2.split("\\s+");

**for**(**int** k = 0; k<strings4.length;k++)

q2.insert(strings4[k]);

String [] strings5 = dataForQ3.split("\\s+");

**for**(**int** k = 0; k<strings5.length;k++)

q3.insert(strings5[k]);

System.***out***.println(s1.size());

System.***out***.println(s2.size());

System.***out***.println(s3.size());

System.***out***.println(q1.size());

System.***out***.println(q2.size());

System.***out***.println(q3.size());

System.***out***.println(s1.equals(s2));

System.***out***.println(s1.equals(s3));

System.***out***.println(s2.equals(s3));

System.***out***.println(q1.equals(q2));

System.***out***.println(q1.equals(q3));

System.***out***.println(q2.equals(q3));

}

}

14

14

17

14

14

17

true

false

false

true

false

false