EECS2030 Day 1 worksheet Thu Sep 7, 2017

See the API attached at the end of this worksheet.

1. **Organization of a Java program**
   (a) What is the package name of the provided API?

   (b) What is the class name of the provided API?

   (c) How many methods appear in the API?

   (d) Can you guess what classes might need to be imported when implementing the class described by the API?

2. **Methods: Basics**
   (a) All of the methods in the API have the same modifiers. What are the modifiers for these methods?

   (b) State the signature for each method in the API.

   signature of `avg`

   signature of `swap2`

   signature of `allGreaterThan`

   signature of `toInt`

   (c) State the return value type for each method in the API.

   return type of `avg`

   return type of `swap2`

   return type of `allGreaterThan`

   return type of `toInt`

(d) All of the following groups of Java statements that are written by a client of the Test2E class contain an error; circle the error and explain what the error is.

i.
```
double avg = Test2E.avg(1.0, 2.0, 3.0);
```

ii.
```
List<Integer> t = new ArrayList<Integer>();
t.add(5);
t.add(6);
List<Integer> u = Test2E.swap2(t);
```

iii.
```
List<Integer> t = new ArrayList<Integer>();
t.add(5);
t.add(6);
List<Integer> u = Test2E.allGreaterThan(t);
```

iv.
```
ArrayList<Integer> t = new ArrayList<Integer>();
t.add(-1);
t.add(0);
double value = toInt(t);
```

3. **Methods: Preconditions and postconditions**

(a) Inspect the API for the method named avg. What are its preconditions? What are its postconditions?

(b) Inspect the API for the method named swap. What are its preconditions? What are its postconditions?

(c) Inspect the API for the method named allGreaterThan. Is "the elements of the list t must be integers" a precondition? Explain why or why not.

4. **Methods: Implementation**

   (a) Implement the method named `avg`.

   (b) Implement the method named `swap`.

   (c) Implement the method named `allGreaterThan`.

5. **Methods: Pass-by-value** Consider the following class having a single method:

```
class Swapper {

    // Swaps the values of a and b
    public static void swap(int a, int b) {
        int tmp = a;
        a = b;
        b = tmp;
    }

}
```

Now consider a client program that tries to use `Swapper`:

```
class Swapper {

    public static void main(String[] args) {
        int x = 99;
        int y = 100;
        Swapper.swap(x, y);
        System.out.println("x = " + x + ", y = " + y;
    }

}
```

(a) What does the program print?

(b) Draw a memory diagram for the client program (ignoring the p̄rintln= statement).

6. **Methods: Javadoc** Complete the Javadoc comments for the following two methods from the API:

(a)

```
/**
 *
 *
 * @param a
 *
 * @param b
 *
 * @param c
 *
 * @return
 */
public static double avg(int a, int b, int c)
```

(b)

```
/**
 * Given a list containing exactly 2 integers, swaps the positions
 * of the integers in the list. For example, given a list
 *
 * <p>
 * <code>[-5, 9]</code>
 *
 * <p>
 * <code>swap2</code> modifies the list so that it becomes
 *
 * <p>
 * <code>[9, -5]</code>
 *
 *
 *
 *
 *
 *
 *
 *
 *
 *
 *
 *
 *
 *
 *
 */
public static void swap2(List<Integer> t)
```

7. **Utility classes**

Create a utility class with the following features:

1. it is located in the package named `eecs2030.test1`
2. its name is `CircleUtil`
3. it has a `public` constant named `TWO_PI` whose value is $2\pi$

4. it has a method named `circumference` that has one parameter of type `double` named `radius` and returns a `double` value
5. the method named `circumference` returns the circumference of the circle having the given radius

Think about what preconditions the method might have.

PACKAGE CLASS USE TREE DEPRECATED INDEX HELP

PREV CLASS NEXT CLASS FRAMES NO FRAMES ALL CLASSES

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

eecs2030.test2

# Class Test2E

java.lang.Object
    eecs2030.test2.Test2E

---

```
public class Test2E
extends Object
```

Test 2 version E.

**Author:**

`EECS2030E Fall 2016`

## *Field Summary*

### Fields

| Modifier and Type | Field and Description |
|---|---|
| `static int` | **`MAX_DIGITS`**<br>The maximum number of digits in a Java int. |

## *Method Summary*

**All Methods** **Static Methods** **Concrete Methods**

| Modifier and Type | Method and Description |
|---|---|
| `static List<Integer>` | `allGreaterThan(List<Integer> t, int max)`<br>Returns a new list containing all of the values in the given list `t` greater than `max`. |
| `static double` | `avg(int a, int b, int c)`<br>Computes the average value of three numbers. |
| `static void` | `swap2(List<Integer> t)`<br>Given a list containing exactly 2 integers, swaps the positions of the integers in the list. |
| `static int` | `toInt(List<Integer> t)`<br>Given a list `t` whose elements are single digits, returns the int value formed by joining the digits. |

### Methods inherited from class java.lang.Object

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

## *Field Detail*

### MAX_DIGITS

```
public static final int MAX_DIGITS
```

The maximum number of digits in a Java int.

**See Also:**

`Constant Field Values`

## *Method Detail*

### avg

```
public static double avg(int a,
                         int b,
                         int c)
```

Computes the average value of three numbers.

**Parameters:**

`a - a number`

`b - a number`

`c - a number`

**Returns:**

`the average of a, b, and c`

### swap2

```
public static void swap2(List<Integer> t)
```

Given a list containing exactly 2 integers, swaps the positions of the integers in the list. For example, given a list

```
[-5, 9]
```

`swap2` modifies the list so that it becomes

```
[9, -5]
```

**Parameters:**

`t - a list containing exactly 2 integers`

**Throws:**

`IllegalArgumentException - if the list does not contain exactly 2 integers`

**Precondition:**

`t is not null`

### allGreaterThan

```
public static List<Integer> allGreaterThan(List<Integer> t,
                                           int max)
```

Returns a new list containing all of the values in the given list `t` greater than `max`. An empty list is returned if no value in `t` is greater than `max`. The list `t` is not changed by this method. For example, if `max == 5` then:

```
 t                     Test2E.allGreaterThan(t, max)
 ---------------------------------------------------
 []                    []
 [4]                   []
 [9]                  [9]
 [4, 5, 6, 7, 8]      [6, 7, 8]
```

**Parameters:**

`t - a list of values`

`max - all values in the returned list will be greater than max`

**Returns:**

`a new list containing all of the values in t that are greater than max`

**Precondition:**

`t is not null`

### toInt

```
public static int toInt(List<Integer> t)
```

Given a list `t` whose elements are single digits, returns the int value formed by joining the digits. The list `t` is not changed by this method. For example, here are some lists and their corresponding int values:

```
 []  (the empty list)          0
 [4]                           4
 [5, 2]                        52
 [8, 7, 3]                     873
 [-1, 0, 0, 0]                -1000
```

If joining the digits of the list produces a positive value greater than `Integer.MAX_VALUE` then `Integer.MAX_VALUE` is returned.

If joining the digits of the list produces a negative value less than `Integer.MIN_VALUE` then `Integer.MIN_VALUE` is returned.

**Parameters:**

`t - a list of digits`

**Returns:**

`the int value corresponding to the digits in t`

**Precondition:**

`t is not null`

**Precondition:**

`the elements of t are integers consisting of exactly one digit`

**Precondition:**

`the first element of t may be negative or positive, but not zero`

**Precondition:**

`all elements except the first are positive or zero`

PACKAGE CLASS USE TREE DEPRECATED INDEX HELP

PREV CLASS NEXT CLASS FRAMES NO FRAMES ALL CLASSES

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD