CMPSC 111
Introduction to Computer Science I
Spring 2018
Janyl Jumadinova

Final Exam Review/Practice Questions

These problems do not represent a structure of the complete final exam. They are also not comprehensive. Instead they are meant for you to use a practice while you review the material for the final exam.

Syntax and Related Errors

These problems deal with basics of program structure.

1. Specify which line(s) of the code below has an incorrect semicolon (an unnecessary one or missing one):

```
1. import java.util.*
2.
3. public class Semi {
5.    public static void main(String[] args); {
7.        int i = 10,
8.            j = 20,
9.            k = 30;
10.        if (k > i+j); {
12.            System.out.println(k);
13.        };
14.        else {
16.            for (int m = 0, m < 10, m++); {
18.                System.out.println(m)
19.            }
20.        }
21.    }
22. }
```

2. Something is wrong with each of these `if` and `if/else` statements. In some cases the errors will cause `javac` to emit error messages and prevent you from executing the program. In others, the program will compile without error, but the effectiveness of the `if` statement is negated by the error. Find the errors and correct them! (NOTE: assume that all variables are properly declared and initialized—the errors lie in the formation of the `if` statements themselves, not elsewhere in the program.)

```
if (i = 3) {
  j = i+i;
}
```

(a)

```
if (i < 3); {
  j = i+i;
}
```

(b)

```
if (i =< 1) {
  j = i+i;
}
```

(c)

```
if (i !> 3) {
  j = i+i;
}
```

(d)

```
if (i != 3) {
  j = i+i;
};
else {
  j = i-i;
}
```

(e)

```
if (i == 3) || (i == 4) {
  j = i+i;
}
```

(f)

```
if (i != 3 || 4) {
  j = i+i;
}
```

(g)

```
if (i && j == 2) {
  i = i+i;
}
```

(h)

3. Something is wrong with each of these loops. In some cases the errors will cause `javac` to emit error messages and prevent you from executing the program. In others, the program will compile without error, but the effectiveness of the loop is negated by the error. (NOTE: assume that all variables are properly declared and initialized—the errors lie in the formation of the loop statements themselves, not elsewhere in the program.)

(a)
```
for (int i == 3; i < 10; i++) {
  sum = sum + i;
}
```

(b)
```
int i = 10;
while (i < 3); {
  sum = sum + i;
}
```

(c)
```
for (int i <= 10) {
  sum = sum+i;
}
```

(d)
```
for (int j = 0, j < 10, j=j+1) {
  sum = sum + j;
}
```

(e)
```
int j = 3;
while (j = 3) {
    sum = sum+j;
    if (sum % 7 != 0)
        j++;
}
```

(f)
```
for (int i = 0; i < 10; i++); {
  j = i+i;
}
```

(g)
```
int j = 10;
while (j >= 10) && (j < 20) {
  j = j+1;
}
```

(h)
```
for (int k = 0; k = 10; k++) {
  System.out.print(k);
}
```

4. Something is wrong with each of the following Java code segments. The errors will prevent `javac` from successfully compiling the program. Find and (when the intention is clear) correct them. (If it is not clear what the programmer intended, just identify the error.)

`String s = "hello";`
`String t = 10;`
`String u = s + t;`
(a)
	`boolean b;`
	`int i = 20;`
	`b = false;`
	`int j = i + b;`
	(b)

`int i = 10, j = 20;`
`i + 10 = j;`
(c)
	`double d = 5;`
	`int k = d;`
	(d)

`int i = 10, j = 10.5, k = 11;` **(e)**	`Scanner scan = new Random();` **(f)**
`char c = "A";` **(g)**	`Random rand = new Random;` **(h)**
`System.out.println("\");` **(i)**	`int single = 1, double = 2, triple = 3;` **(j)**

Classes, Instance Variables, Methods

6. Write "get" and "set" methods for each instance variable shown below. "get" methods should simply return the value of the appropriate instance variable; "set" methods should have a single parameter whose value should be stored in the appropriate instance variable.

```
  ...
public int x;
public double y;
public String z;
public boolean b;
  ...
```

7. (a) Suppose the instance variables in the preceding problem belong to a class named "`Thing`". Write a four-parameter constructor that initializes the instance variables to values provided in the parameters; order the parameters in the same order as the variables were declared in the previous question.

   (b) Write a statement to create a new `Thing` object with initial values of 17 (for `x`), 5.5 (for `y`), `"Hello"` for `z`, and `true` for `b`.

8. Here is a complete class named `Tree`:

```
import java.util.*;
public class Tree {
    public String name;
    public boolean deciduous;

    public Tree(String n, boolean d) {
        name = n;
        deciduous = d;
    }

    public String getName() {
        return name;
    }

    public boolean isDeciduous() {
        return deciduous;
    }
}
```

Write a `main` method that creates three `Tree` variables named `"oak"`, `"elm"`, and `"pine"`. The first two are deciduous, the last is not. That's all—the method just declares three variables and doesn't do anything else.

9. Here are three classes (only the method names are shown; you should assume that the code inside the methods is not relevant to this question).

   Write the statement in the `main` method to create a new `Alpha` variable named `a` with number 42 and type `"large"`. Then create a new `Beta` variable named `b` that contains `a`. Finally, write a print statement to display information about variable `b`'s `alf` value.

```
public class Alpha {
  private int number;
  private String type;

  public Alpha(int n, String t) {
    ...
  }

  public String toString() {
    ...
  }
}
```

```
public class Beta {
  private Alpha alf;

  public Beta(Alpha a)   {
    ...
  }

  public Alpha getAlf() {
    ...
  }
}
```

```
public class Main {
  public static void main(
                String[] args)  {
      // SEE QUESTION 9
  }
}
```

10. What is the final value of `count` when each of the following loops is finished executing?

(a)

```
int count = 0;
for (int j = 1; j < 10; j++) {
  count = count + 1;
}
```

(b)

```
int count = 0;
for (int j = 0; j <= 10; j++) {
  count = count + 1;
}
```

(c)

```
int count = 0;
int value = 1;
while (value < 5) {
  value = value + 1;
  count = count + value;
}
```

(d)

```
int count = 0;
int j = 1;
while (j <= 10) {
  count = count + 1;
  j = 2 * j;
}
```

(e)

```
int count = 0;
int j = 5;
while (j >= 0) {
  count = count + 1;
  j--;
}
```

(f)

```
int count = 0;
int value = 1;
while (count < 10) {
  count = count + value;
  value = value + 1;
}
```

(g)

```
int count = 0;
for (int j = 0; j < 10; j = j + 2) {
  count = count + 1;
}
```

(h)

```
int count = 0;
for (int j=1; j <= 10; j = 2*j) {
  count = count + 1;
}
```

11. What are the final values of variables `a` and `b` in each of the following Java code segments? For each one, assume the initial values of `a` and `b` are given by

```
int a = 10, b = 20;
```

(a)

```
if (a < 2*b) {
   a = a + b;
}
else {
   b = b - a;
}
```

(b)

```
if (2 * a == b) {
   b = a + b;
}
else {
   a = a - b;
}
```

(c)

```
if (a == b - 10) {
   a = b;
   b = 15;
}
if (a > b) {
   b = a;
   a = 5;
}
```

(d)

```
if ((a == 10 && b == 20) || a > b) {
   b = 2 * b;
   a = a - 5;
}
if (b > 20 || a < 5) {
    a = 0;
}
```

(e)

```
if (b == 20 && ! (a < b)) {
   b = a;
}
a = 20;
```

(f)

```
if (a > b || 2*a == b) {
   b = b + a;
}
a = a + b;
```

12. Write the Java statements needed to find the sum of the first 20 positive odd integers, that is, the sum $1 + 3 + 5 + \cdots$ (20 terms in the sum). The final value should be stored in an `int` variable named `sum` which you must declare and initialize. Use some kind of loop structure; don't simply type in all of the odd integers in a great big sum expression!

13. Suppose an `int` variable named `x` has been initialized to some unspecified positive value. Write the Java statements needed to determine whether or not `x` satisfies the following condition:

    either `x` is divisible by 3 *or* `x` is between 10 and 20 inclusive (or both)

    Print the word "`yes`" if `x` satisfies the condition, "`no`" otherwise. For instance, values such as 1, 2, 5, 23, and many others will produce a "`no`" output, while values such as 3, 6, 10, 12, 20, 30, and many others will produce a "`yes`" output.

14. Assume `scan` is a `Scanner` variable that has already been declared and initialized. Write the Java statements needed to input a sequence of `double` values and determine how many of them are strictly greater than 10.0. The sequence ends when the user enters a value that is

less than or equal to 0.0. The final count should be stored in an `int` variable named `count`. You should declare and initialize `count` and any other variables you need (but not `scan`, which is already given).

Built In Classes

15. What are the values of the following expressions (note that some values may be `String`, `int`, `char`, or `boolean`)? Assume that we have the following variables declared:

```
String s1 = "final", s2 = "exam";
```

`s1 + s2` **(a)**	`s1.substring(1,3)` **(b)**
`s1 + s2.substring(2)` **(c)**	`(s1 + s2).length()` **(d)**
`s1.charAt(3) == s2.charAt(2)` **(e)**	`"abc" + s2 + s1.length()` **(f)**

16. Assume `rand` is a variable of class `Random` that has been declared and initialized.

    (a) What are the smallest and largest possible values produced by the following expression?

    ```
    rand.nextInt(10)-5
    ```

    (b) List *all* the possible values of the following expression:

    ```
    "abcdefg".charAt(rand.nextInt(3)+1)
    ```

    (c) Write an expression that generates a random *even* integer between 0 and 10, inclusive.

    (d) Write an expression that generates a random `double` value between -10.0 and 10.0. (The values should not be limited to just a small subset of regularly-spaced double values such as -10, -9.5, -9, etc. Your expression should be capable of generating *any* value between -10 and 10.)

17. Suppose `scan` has been declared as a variable of class `Scanner` and has been properly initialized to accept keyboard input. Write the Java statements needed to input and save an `int`, a `String` containing no blanks or other white space, and a `double` value, in that order. You will need to declare variables to hold the three values.

Arrays and Array Lists

18. Study the following Java code, then answer the questions below.

```
import java.util.*;
...
ArrayList<String> sList = new ArrayList<String>();
sList.add("bat");
sList.add("cat");
sList.add("dog");
sList.add("frog");
...
```

(a) What is the value of `sList.get(3)`?

(b) What is the value of `sList.size()`?

(c) What is the value of `sList.contains("at")`?

(d) What is the value of `sList.get(0).contains("at")`?

(e) What is the final value of the variable `m` in the following code:

```
  ...
String m = "";
for (int i = 0; i < sList.size(); i++) {
    String a = sList.get(i);
    if (a.contains("g")) {
        m = m + a;
    }
}
  ...
```

(f) Write a loop in Java to print out all values in `sList` in reverse order, i.e., from last item to first item.

19. Write the Java statements needed to create a new `ArrayList` containing integer values and to fill it with the first 1000 consecutive integers, starting with 0.

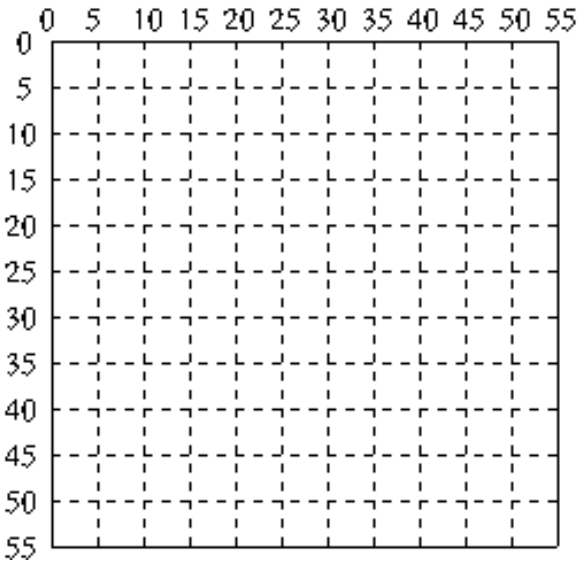
20. Given the following Java declaration:

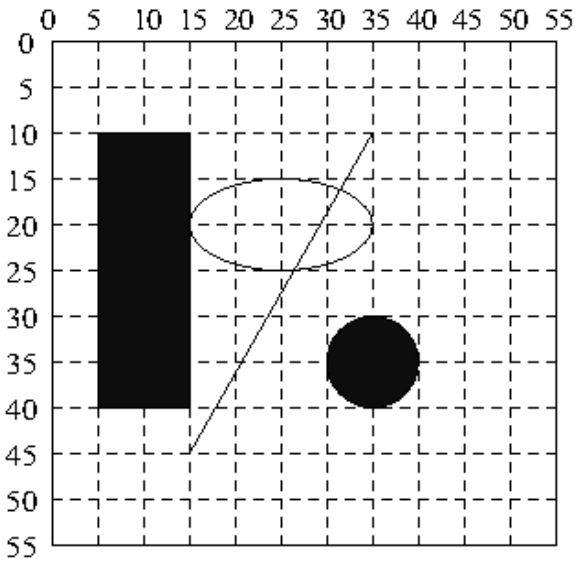
```
int x[] = new int[25];
```

Write the Java statements needed to fill `x` with the values of the first 25 positive multiples of 3, that is, 3, 6, 9, ...

Graphics

21. For part (a), draw the image corresponding to the Java code. For part (b), write the Java code corresponding to the image. In each part, assume that the `Graphics` object is named `page`. Just lightly scribble to show filled objects—don't worry about carefully shading in every little pixel!






```
page.fillRect(15,30,30,15);
page.drawLine(5,10,30,40);
page.drawOval(10,40,30,20);
```

(a) **(b)**

22. Consider the three Java programs shown in the figure on the following page, together with the image produced by running `Draw.java`.

    (a) To add a third tree to the image (any location you wish), what statements would you add to the program and where would you put them?

    (b) Suppose we wish to place tree `t1` at a *random* location inside the 300-by-300 window. What statements would you add to the program and where? (Include any new variables and/or `import` statements.)

    (c) If we wanted to create an array `t` of five `Tree` objects somewhere inside the `Forest` class, how would we declare and initialize the variable `t`? (I don't care where the declaration and initialization take place—it could be an instance variable or a local variable. I'm only interested in the form of the declaration and initialization.)

```
import javax.swing.*;

public class Draw {
  public static void main(String[] args) {
    // WINDOW TO HOLD THE "Forest":
    JFrame window = new JFrame("Forest");
    Forest f = new Forest();
    window.getContentPane().add(f);
    window.setDefaultCloseOperation(...
    window.pack();
    window.setVisible(true);
  }
}
```

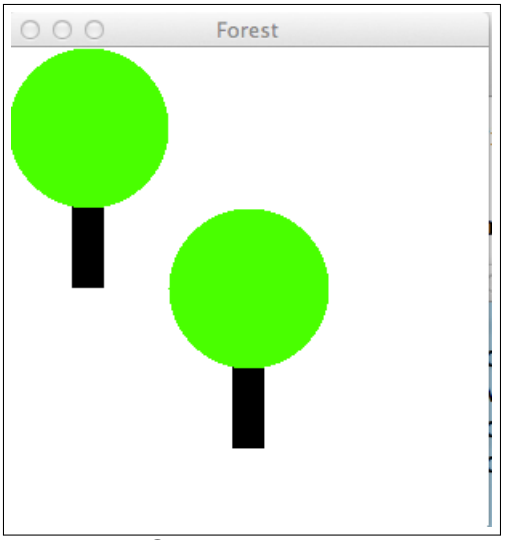
Draw.java

```
import java.awt.*;

public class Tree {
  // INSTANCE VARS:
  private int x,y;
  // CONSTRUCTOR
  public Tree(int x, int y) {
    this.x = x;
    this.y = y;
  }

  // DRAWS A TREE AT POSITION (x,y):
  public void draw(Graphics page) {
    page.setColor(Color.black);
    page.fillRect(x+40,y+50,20,100);
    page.setColor(Color.green);
    page.fillOval(x,y,100,100);
  }
}
```

Tree.java

```
import java.awt.*;
import javax.swing.*;

public class Forest extends JPanel {
  // NO INSTANCE VARIABLES!

  // CONSTRUCTOR SETS SIZE
  public Forest() {
    setPreferredSize(new Dimension(300,
                                   300));
  }

  // DRAW TWO TREES:
  public void paintComponent(Graphics
                             page) {
    // clear the window:
    page.setColor(Color.white);
    page.fillRect(0,0,300,300);

    // define two trees:
    Tree t1 = new Tree(0,0);
    Tree t2 = new Tree(100,100);

    // draw them:
    t1.draw(page);
    t2.draw(page);
  }
}
```

Forest.java



Output

Miscellaneous

23. Define the following terms:

    (a) algorithm
    (b) compiler
    (c) concatenation
    (d) `static` variable
    (e) binary
    (f) wrapper class
    (g) instantiation
    (h) boolean operator
    (i) casting operator