CIS 110 Exam 1, Summer 2021 **ANSWERS**

Q1. Academic Integrity

By copying my name below, I certify that I will complete this exam in compliance with the Academic Integrity policies of CIS 110 and the University of Pennsylvania. In particular, my answers will reflect only my own work and during the exam I will only access acceptable materials including my own notes, my previous work in this class, Piazza, or the course website.

eg. *Shivin Uppal*

Q2. Reviewing the Building Blocks of Code

1. True or False? Any `char` value can be stored in an `int` variable, but not all `int` values can be stored in a `char` variable. **True**
2. Fill in the most appropriate word. If code would result in an error, write "ERROR". `_____ x = 2.0 / (1 / 2);` **`double`**
3. True or False? A chain of `if`/`else if` statements must finish with an `else` statement. **False**
4.

```
public static double calculator(double a, double b, String operation) {
    double result = 0;
    switch(operation) {
        case "plus":
            result = a + b;
        case "minus":
            result = a - b;
        case "times":
            result = a * b;
        case "div":
            result = a / b;
    }
    return result;
}
```

If the line of code `double x = calculator(8.0, 2.0, "minus");` is executed, what will `x`'s value be, and why will it be this value? **`4.0`, because the `break;` statement is missing in each case, so every case after the `"minus"` one will be evaluated. The last case evaluated would be `"divide"`, so `result` will be `4.0`.**

5. True or False? Any code written using a `for` loop can be rewritten to use a `while` loop. **True**
6. Fill in the blanks below so the loop does not run infinitely

```
for (int i = 10; i > 0; ____) {
    System.out.println("Hello World");
}
```

`i--` or `i /= 2` or anything to that effect which leads to `i` being reduced to `0` or less.

7. True or False? The line of code `PennDraw.enableAnimation(60);` makes the program wait for 60 seconds before it is allowed to continue executing code when it invokes `PennDraw.advance();` **False**

8. In one line of code draw the image shown below, assuming no other PennDraw functions have been called beforehand: `PennDraw.filledCircle(0.75, 0.75, 0.25);`

9. True or False? As long as functions are declared inside the same class, they can access variables declared inside one another. **False**

10. The following snippet represents a function with most of its signature missing. Fill in the blank to specify the function's return type, name, and input argument(s) so that the program will compile. If a variable is used but not declared in the function body, you can assume that it is an input argument to the function. Make sure to use good style and give the function a name that is related to its behavior.

```
public static _______ _________(________ ________) {
    double total = 0;
    for (int i = 0; i < arr.length; i++) {
        total += arr[i];
    }
    return total / arr.length;
}
```

`public static double avgOfArray(double[] arr)` or `public static double avgOfArray(int[] arr)` or any other relevant function name.

Q3. The Fastest Cats

A. `String`

B. `String`

C. `3`

D. `int i = 0; i < results.length; i++`

E. `second`

F. `second`

G. `first`

H. `results[i]`

I. `second`

J. `third`

K. `results[i] == first`

L. `0`

M. `fastest`

Q4. The Great Cat Race

```
public static int[] intervalTimeGenerator(int[] finalTimes) {

    int[] intervalTimes = new int[finalTimes.length];
    intervalTimes[0] = 0;

    for (int i = 1; i < finalTimes.length; i++) {
        intervalTimes[i] = finalTimes[i] - finalTimes[i - 1];
    }

    return intervalTimes;
}
```

Q5. Neko Atsume

**Note that the question asked for 7 errors, but there were actually 8 errors. As a result, you received credit for identifying any 7 out of the 8 bugs.

```
public class NekoAtsumeBonus { // Error 0: Shouldn't have () after Class
Name
    public static void main(String[] args) { // Error 1: Should be
String[] not String
        int numCats = Integer.parseInt(args[0]); // Error 2: Need to parse
args[0] to int

        int[] catCollection = new int[numCats];

        for(int i = 0; i < catCollection.length; ++i) {
            catCollection[i] = (int) (Math.random() * 100); // Error 3:
Need to cast double to int
        }
```

```
        int catEvenStripeSum = sumOfEvenStripes(catCollection);
    }

    public static int sumOfEvenStripes(int[] arr) {
        int len = arr.length;
        int sum = 0;
        for(int i = 0; i < len; ++i) { // Error 4: Should be < len instead
of <= len
            if(arr[i] % 2 == 1) { // Error 5: Should be arr[i] instead of
i
                continue; // Error 6: Should be continue instead of break
            }
            sum = sum + arr[i];
        }
        return sum; // Error 7: Should return instead of print
    }
}
```

Q6. Mauby and Dudu in the Living Room

```
public static boolean didCatSpendTooLong(String[] log, String cat) {
    int total = 0;
    for (int i = 0; i < log.length; i++) {
        if (log[i].equals(cat)) {
            total++;
            if (i < log.length - 2 && log[i].equals(log[i + 1]) &&
log[i].equals(log[i + 2])) {
                return true;
            }
        }

        if (total > 10) {
            return true;
        }
    }

    return false;
}
```

Alternate Solution

```
public static boolean didCatSpendTooLong(String[] log, String cat) {
    int total = 0; // total hours spent by the cat
    int consecutive = 0; // consecutive hours spent by the cat
    for (int i = 0; i < log.length; i++) {
        if (log[i].equals(cat)) {
            total++;
            consecutive++;
```

```
            if (consecutive >= 3) {
                return true;
            }
        } else {
            // if some other cat is in the room, reset consecutive to 0
            consecutive = 0;
        }
    }
    // you reach here only if the cat didn't spend >= 3 consecutive hours
    return total > 10; // check if spent > 10 hours total
}
```

```
public static boolean didSameCatDoSunriseAndSunset(String[] log)  {
    return !(log[5].equals("E")) && log[5].equals(log[20]);
}
```

```
public static boolean didAnyCatAlternate(String[] log) {
    for (int i = 0; i < log.length - 2; i++) {
        if (log[i].equals("M")) {
            if (log[i + 1].equals("D") && log[i + 2].equals("M")) {
                return true;
            }
        } else if (log[i].equals("D")) {
            if (log[i + 1].equals("M") && log[i + 2].equals("D")) {
                 return true;
            }
        }
    }
    return false;
}
```

Alternate Solution

```
public static boolean didAnyCatAlternate(String[] log) {
    for (int i = 0; i < log.length - 2; i++) {
        if (log[i].equals(log[i + 2]) && !log[i].equals(log[i + 1]) &&
!log[i].equals("E") && !log[i+1].equals("E")) {
            return true;
        }
    }
    return false;
}
```

```
public static String findTheHappierCat(String[] log) {
    int mauby = 0;
    int dudu = 0;

    if (didCatSpendTooLong(log, "M") || didCatSpendTooLong(log, "D") ||
            didSameCatDoSunriseAndSunset(log) || didAnyCatAlternate(log))
{
        return "None";
    }

    for (int i = 0; i < log.length; i++) {
        if (log[i].equals("M")) {
            mauby++;
        }
        else if (log[i].equals("D")) {
            dudu++;
        }
    }

    if (mauby > dudu) {
        return "Mauby";
    } else if (dudu > mauby) {
        return "Dudu";
    } else {
        return "Tie";
    }

}
```