CIS 1100 — Spring 2024 — Exam 1

Full Name: ____

PennID (e.g. 80472698): ____

"My signature below certifies that I have complied with the University of Pennsylvania's Code of Academic Integrity in completing this exam."

Signature ____

Instructions are below. not complying will lead to a 0% score on the exam.

- Do not open this exam until told by the proctor.
- You will have exactly 60 minutes to take this exam.
- There is scratch space at the end of the exam.
- Make sure your phone is turned OFF (not on vibrate!) before the exam starts.
- Food and gum are not permitted—don't be noisy or messy.
- You may not use your phone or open your bag for any reason, including to retrieve or put away pens or pencils, until you have left the exam room.
- This exam is closed-book, closed-notes, and closed computational devices.
- If you get stuck on a problem, it may be to your benefit to move on to another question and come back later.
- All code must be written in proper Java format, including all curly braces and semicolons.
- Do not separate the exam pages. Do not take any exam pages with you. The entire exam packet must be turned in as is.
- Only answers on the FRONT of pages will be graded. There are two blank pages at the end of the exam if you need extra space for any graded answers.
- Use a pencil, or blue or black pen to complete the exam.
- If you have any questions, raise your hand and a proctor will come to you.
- When you turn in your exam, you may be required to show your PennCard. If you forgot to bring your ID, talk to an exam proctor immediately.
- We wish you the best of luck!

Q1: Types Fill in the Blank

In the column marked **“Type,”** choose the type (e.g. int, char, etc.) for the variable that would allow the line to compile, or write **“compilation error”** if there is an error in the expression that makes its type undefined. You do not need to write the value of the expression.

Statement	Type
`______ x = "cis".equals("cs");`	boolean
`______ x = "code".charAt(3) == "d";`	CE
`______ x = 5 + Integer.parseInt("34");`	int
`______ x = new int[13];`	int[]
`______ x = 0 <= Math.random() < 1;`	CE
`______ x = "caesa" + 'r';`	String
`______ x = 4 + (28 / (8 / 2.0));`	double

Q2: Values Fill in the Blank

Write the value that gets printed, or write **“runtime error”** if there is an error during the execution of these lines of the program. Each code snippet is independent of the others

Question 2.1

```
System.out.println(3 + Integer.parseInt("2" + "1"));
```

Answer: 24

Question 2.2

```
int x = 5;
int y = 2;
System.out.println(x / y);
```

Answer: 2

Question 2.3

```
int[] arr = {2, 4, 6, 8};
arr[arr.length - 1] = arr[arr.length - arr[1]] * arr[2];
System.out.println(arr[3]);
```

Answer: 12

Question 2.4

```
int a = 12;
String s = "4.0"
System.out.println(a / Double.parseDouble(s));
```

Answer: 3 (or error due to missing semicolon)

Question 2.5

```
// Math.random() evaluates to a random double between 0 and 0.999...
System.out.println((int) Math.random());
```

Answer: 0

Question 2.6

```
String a = "midterm";
int i = a.length();
System.out.println(a.charAt(i));
```

Answer: runtime error

Q3: Tracing

Here's a class that features a few functions.

```
public class TracingExercise {
  public static boolean mysteryOne(String word) {
    boolean check1 = word.charAt(0) == 'A';
    boolean check2 = word.charAt(word.length() - 1) == 'w';
    boolean check3 = word.length() > 6;

    return (check1 || check2) && check3;
  }

  public static int mysteryTwo(String word, int x, int y) {
    boolean check = mysteryOne(word);
    int value = x * y;

    if (!check && value < 15) {
        return mysteryThree(value, word.length());
    } else {
        return 1 + mysteryThree(x + x, y + y);
    }
  }

  public static int mysteryThree(int a, int b) {
    int value1 = a % b;
    int value2 = a / b;
    value2--;

    return value1 + value2;
  }

  public static void main(String[] args) {
    System.out.println(mysteryTwo("Arrows", 2, 4));
    System.out.println(mysteryTwo("Sparrow", 7, 2));
    System.out.println(mysteryTwo("wow", 4, 3));
    System.out.println(mysteryTwo("alive", 9, 6));
  }
}
```

When the program is run as `java TracingExercise`, four lines are printed. Fill in the blanks in the "Printed Line" column (on the next page) to show what values get printed.

Please write your solution to Q3 on this page.

Print Statement	**Printed Line**
`System.out.println(mysteryTwo("Arrows", 2, 4));`	2
`System.out.println(mysteryTwo("Sparrow", 7, 2));`	5
`System.out.println(mysteryTwo("wow", 4, 3));`	3
`System.out.println(mysteryTwo("alive", 9, 6));`	7

Spring 2024 Exam 1 Answer Key

Problem 4:

```java
public class Outfit {
    public static void main(String[] args) {
        // starter arrays with individual clothes
        String[] bhrajitShirts = { "yellow", "black", "button down brown", "denim" };
        String[] priyaShirts = { "red", "sparkly mesh", "black tank", "fruits", "purple" };

        // combined array initialization
        String[] combinedWardrobe = new String[bhrajitShirts.length + priyaShirts.length];
        for (int i = 0; i < combinedWardrobe.length; i++) {
            // fill up array first with Bhrajit's clothes
            if (i < bhrajitShirts.length) {
                combinedWardrobe[i] = bhrajitShirts[i];
                // after Bhrajit's clothes, fill up with Priya's clothes
            } else {
                combinedWardrobe[i] = priyaShirts[i - bhrajitShirts.length];
            }
        }
        // Select ANY valid index from the combinedWardrobe array twice.
        int bhrajitIndex = (int) (Math.random() * (combinedWardrobe.length));
        int priyaIndex = (int) (Math.random() * (combinedWardrobe.length));
        if (priyaIndex == bhrajitIndex) { // check if both selected the same index
            priyaIndex = (priyaIndex + 1) % combinedWardrobe.length; // if so, pick the next index
        }
        String bhrajitFit = combinedWardrobe[bhrajitIndex];
        String priyaFit = combinedWardrobe[priyaIndex];

        System.out.println(bhrajitFit);
        System.out.println(priyaFit);
    }
}
```

Problem 5:

```
public class HikePlanning {
    public static boolean isExciting(double[] hike) {
        double max = hike[0];
        double firstElevation = hike[0];
        for (int i = 1; i < hike.length; i++) {
            if (hike[i] > max) {
                max = hike[i];
            }
        }
        return max >= firstElevation * 2;
    }

    public static int firstDramaticSegment(double[] hike) {
        double current = hike[0];
        for (int i = 1; i < hike.length; i++) {
            if (Math.abs(hike[i] - current) > 1000) {
                return i - 1;
            }
            current = hike[i];
        }
        return -1;
    }

    public static boolean likeHike(double[] hike) {
        return isExciting(hike) && firstDramaticSegment(password) >= 1;
    }
}
```