COMP110 – WS3 KEY Name:_____ PID:_____

1. True/False

__F_ a. A function can either return a value or print a value but it cannot do both.
__F_ b. A function can return more than one expression during a single invocation of that function.
__T_ c. A function can print more than one expression during a single invocation of that function.
__F_ d. Any time a function returns an expression, that expression is always printed to the webpage.
__F_ e. A class must be defined in the main function.
__T_ f. The dot operator is used to access an object's property.
__T_ g. Classes define new data types.
__T_ h. The new keyword is used when constructing an object.

2. Given the following class declaration, answer the questions that follow.

```
class UNCStudent {
    major: string = "Undeclared";
    creditHours: number = 0;
    preMed: boolean = false;
    gpa: number = 0;
}
```

a. Write a line of code to declare and initialize a new `UNCStudent` object named `roy`.

```
let roy: UNCStudent = new UNCStudent();
```

b. Change the `major` property of your `roy` object to be a new major, and change the `gpa` property to a new gpa.

```
roy.major = "Computer Science";
roy.gpa = 3.5;
```

c. Write an if-then statement to change roy's creditHours property to 100 if his preMed property is true, and to 50 otherwise.

```
if (roy.preMed) {
    roy.creditHours = 100;
} else {
    roy.creditHours = 50;
}
```

d. Assume that your previous code was written and placed into a main method to run. Fill out the table below with the value of each of roy's properties after your code has run.

Property	Value
major	"Computer Science"
creditHours	50
preMed	false
gpa	3.5

3. Given the function below, answer the questions on the right. For part A, fill out the table below with values for `a`, `b`, and `c` such that when `random(a,b,c)` is called, the value in the leftmost column is returned. If the return statement is unreachable, write "N/A" in the boxes for `a`, `b`, and `c`.

```
let random = (a: number, b: number, c: number): string => {
    if (a > b) {
        if (b > c) {
            return "c";
        } else {
            return "o";
        }
    }
    if (b < c) {
        if (a === b && c < a) {
            return "m";
        }
        return "p";
    }
    if (a === b && b === c) {
        return "!";
    }
    return "woo";
};
```

A..

return:	a	b	c
"p"	1	2	3
"c"	3	2	1
"!"	1	1	1
"woo"	1	3	2

B. Underline any unreachable code. If there is none, write "N/A".

C. What would be the ***output*** if `random(1,1,1)` were called within the main function?

Nothing. There is no output because `random` ***returns*** a value but it does not ***print*** anything.

4. For the following question, your goal is to write the code in the `main` function that yields the desired output. This output requires exactly 4 lines of code. Write each of the 4 lines of code in the correct order on the right.

```
let bar = (x: string, num: number): string => {
   if (num === 0) {
       print("done");
       return "finished";
   } else {
       print("recurring: " + x);
       return bar(x.substr(1), num - 1);
   }
};
let foo = (x: string): string => {
   print(x + x);
   print("finished");
   return x + x + x;
};
let main = async () => {
  // Your code goes here
};
main();
```

Desired output:

fastfast
string

finished
string

fastfastfast
string

recurring: rat
string

recurring: at
string

done
string

catcat
string

finished
string

main
string

Line 1:

```
print(foo("fast"));
```

Line 2:

```
bar("rat", 2);
```

Line 3:

```
foo("cat");
```

Line 4:

```
print("main");
```

5. Answer the questions below using the following classes:

```
class CoffeeBlend {
    caffeinePerOunce: number = 0; // in milligrams
    flavor: string = "";
}

class CupOJoe {
    ounces: number = 0;
    caffeineContent: number = 0; // in milligrams
    flavor: string = "";
}
```

a) Write a function named `brewCoffee` that has two parameters -- a `CoffeeBlend` object named `blend` and a `number` named `ouncesToBrew` (representing the number of ounces to brew) -- and returns a `CupOJoe` object made using that blend. This means that the `CupOJoe` object you return should match the `flavor` of the blend, contain the number of ounces given by `ouncesToBrew`, and should have a *total* `caffeineContent` derived from `ouncesToBrew` and the `CoffeBlend`'s `caffeinePerOunce`.

```
let brewCoffee = (blend: CoffeeBlend, ouncesToBrew: number): CupOJoe => {
   let cup: CupOJoe = new CupOJoe();
   cup.ounces = ouncesToBrew;
   cup.caffeineContent = blend.caffeinePerOunce * ouncesToBrew;
   cup.flavor = blend.flavor;
   return cup;
};
```

b) Computer scientists need *a lot* of caffeine to function properly. Write a function named `lotsaCaffeine` with one parameter -- a `List` of `CoffeeBlend` objects named `blends` -- that finds the `CoffeeBlend` object in the list with the *largest* `caffeinePerOunce` value and returns a 72oz `CupOJoe` object made using that `CoffeeBlend` (hint: you should be using the `brewCoffee` function you defined in (a)). For the base case, simply return a `CupOJoe` object with the default property values.

```
let lotsaCaffeine = (blends: List<CoffeeBlend>): CupOJoe => {
   if (blends === null) {
       return new CupOJoe();
   }
   let currentCup: CupOJoe = brewCoffee(first(blends), 72);
   let maxFromRestOfList: CupOJoe = lotsaCaffeine(rest(blends));
   if (maxFromRestOfList.caffeineContent > currentCup.caffeineContent) {
       return maxFromRestOfList;
   } else {
       return currentCup;
   }
};
```

6. Based on the following Book class and foo function, answer the questions below.

```
class Book {
  title: string = “”;
  author: string = “”;
  page: number = 0;
}
let foo = (title: string, author: string, currentPage: number): number => {
  if (title.includes(“in”) && author.includes(“er”)) {
    currentPage = currentPage + currentPage;
  } else {
    currentPage = currentPage + 1 * 2;
  }
  currentPage = 1 + currentPage;
  print(currentPage + 1);
  return currentPage;
};
let main = async () => {
  let myBook = new Book(); // (1)
  myBook.title = await promptString(“Give me a title”); // (2)
  myBook.author = await promptString(“Give me an author”); // (3)
  myBook.page = await promptNumber(“Give me my current page”); // (4)
  let a: number = foo(myBook.title, myBook.author, myBook.page); // (5)
  let yourBook = new Book();
  yourBook.title = “Breaking Dawn”;
  yourBook.author = “Stepanie Meyer”;
  yourBook.page = 24;
  let b: number = foo(yourBook.title, yourBook.author, yourBook.page); // (6)
  if (a > b) {
    print(“My book is better than yours”);
  } else {
    print(“We are reading good books”);
  }
};
main();
```

a. For all of the following questions, suppose the user inputs “Mockingjay” , “Suzanne Collins” , and 15 for the lines denoted by (2), (3), and (4). Write below all printed output after main finishes executing.

19
50
We are reading good books

b. What is the value of a after the line denoted by (5) is executed?

18

c. What is the value of b after the line denoted by (6) is executed?

49

d. What are the values of the properties of myBook after main finishes execution? Please write them in the following format: <name> = <value>

title = “Mockingjay”
author = “Suzanne Collins”
page = 15