COMP110 - Worksheet 00 Name: _____ PID: _____

1. (a) F The three primitive data types in TypeScript are `number`, `boolean`, and `character`.
   (b) F All functions must have a `return` type.
   (c) T Parameters are local variables within the function body's block meaning their names are only accessible inside the function.
   (d) F When the `+` operator is used with one or more numbers, the process is called concatenation.
   (e) T In order to use a function, there must be a call to that function elsewhere in the code.
   (f) T A function call is **always** complete as soon as any `return` statement is reached.
   (g) F Arguments are specified within the parentheses of a function definition.
   (h) F Variable names can contain spaces.

2. Fill in each column with the value of the variable after each line of code has run. If the variable has not been declared, write ?. If the variable has been declared but not initialized, write *.

	x	y	z
let x: number = 15;	15	?	?
let y: number;	15	*	?
x = x + 8;	23	*	?
let z: number = 9;	23	*	9
y = x + z;	23	32	9

3. Given the incomplete functions to the left and the function bodies to the right, write the letter of the function for which each body could be used (with no errors). A body can work with 0, 1, or more function definitions.

```
let a = (x: number): number => {
    // ???
};
let b = (x: boolean): boolean => {
    // ???
};
let c = (x: boolean): number => {
    // ???
};
let d = (x: string): boolean => {
    // ???
};
let e = (x: number): string => {
    // ???
};
let f = (x: number): boolean => {
    // ???
};
let g = (x: string): string => {
    // ???
};
```

(a) a, b, g `return x;`

(b) b, d, f `return true;`

(c) a `return x * 2;`

(d) f `return x < 10;`

(e) a, g `return 7 + x;`

(f) b `return x && true;`

(g) a, c `return 5;`

(h) e, g `return x + "2";`

4. Write a function called `repeatBeat` that takes in a string `str` and a number `n` and returns the string repeated n times with a space between each repeat. You may assume `n >= 0`. There are some sample inputs and outputs below.
Note: To receive full credit, there can be no space at the end of the output.

`repeatBeat("whoop", 3);` `// returns "whoop whoop whoop"`	`repeatBeat("swoosh", 0);` `// returns ""`

```
//Write your code here
// there is more than one solution, this is just one
export let repeatBeat = (str: string, num: number): string => {
    let output =
    while (num > 1) {
```

5. Write a function named `modulo` that takes in 2 number parameters, named `x` and `y`, and returns the result `x % y`. You may assume that both `x` and `y` are positive.
Note: In order to receive any credit, you may not use the modulo operator, and you must use a while loop.

```
// this is just one solution
export let modulo = (x:number, y:number) => {
    while (x >= y) {
        x = x - y;
    }
    return x;
};
```

6. Given the two functions on the left, answer the questions in the box on the right.

```
let isEven = (x: number): boolean => {
    return x % 2 === 0;
};

let getName = (y: number): string => {
    if (isEven(y)) {
        return "Michael";
    } else {
        if (y === 4) {
            return "Dwight";
        } else {
            if (y === 3) {
                return "Jim";
            }
            return "Stanley";
        }
    }
};
```

(a) What will be returned by following function call: `getName(3);`

"Jim"

(b) What will be returned by the following function call: `getName(4);`

"Michael"

(c) If there is any unreachable code (i.e. code that can never be reached in any circumstance), circle it. If there is not, write "NA" below.

(d) Make a function call that will result in the value `"Stanley"` being returned.

getNone(5);
(odd number other than 3)

7. Given the function `foo` on the left, determine the values of the function calls on the right.

```
let foo = (n: number): number => {
  let x = 0;
  while (n > 0) {
    let r = n % 10;
    x = x + r;
    r = r / 10;
    n = n / 10 - r;
  }
  return x;
}
```

(a) What will be returned by the following function call: `foo(12);`

3

(b) What will be returned by the following function call: `foo(342);`

9

(c) What will be returned by the following function call: `foo(327);`

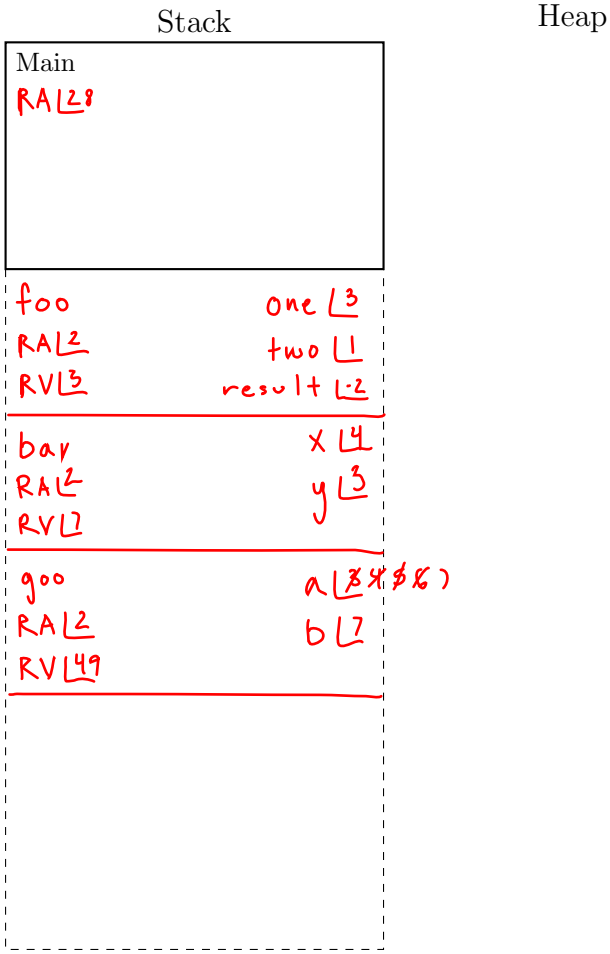
12

(d) In one sentence, what does the `foo` function do?

returns sum of digits in a number

1. Given the code listing below, draw an environment diagram **paused at the moment line 3** is reached. In each frame on the stack, remember to include a space for the return address and return value, if any.

```
1  let main = async() => {
2      print(goo(foo(3, 1), bar(4, 3)));
3      // Pause here
4  };
5
6  let foo = (one: number, two: number): number => {
7      let result = two - one;
8      return one;
9  };
10
11 let bar = (x: number, y: number): number => {
12     if (x % y === 0) {
13         print("Go");
14     } else {
15        print("Heels");
16     }
17     return x + y;
18 };
19
20 let goo = (a: number, b: number): number => {
21     while (a < b) {
22         print("ayooo");
23         a++;
24     }
25     return a * b;
26 };
27
28 main();
```



The following questions are about the state of the program at the moment the evaluation is **paused at line 3**. If a question is asked about a name that has not been initialized, then respond with `undefined`.

1.1 From `goo`'s frame, what is the value of `a`? **7**

1.2 From `goo`'s frame, how many times do you iterate through the while loop? **4**

1.3 Write a function call to `bar` that results in `"Go"` being printed. **bar(4,2)**

1.4 From `foo`'s frame, what is the value of `result`? **-2**

1.5 What is the printed output of this program once it completes?

Heels ayooo ayooo ayooo ayooo 49