COMP110 - Optional Worksheet 02 Name: _____ PID: _____

1. In the space below, create a class called `Student` with the following properties: `name`, which is a `string`; `gradYear`, which is a `number`; `age`, which is a `number`; and `isUndergrad`, which is a `boolean`. Initialize each of the properties with your own name, graduation year, age, and undergraduate status.

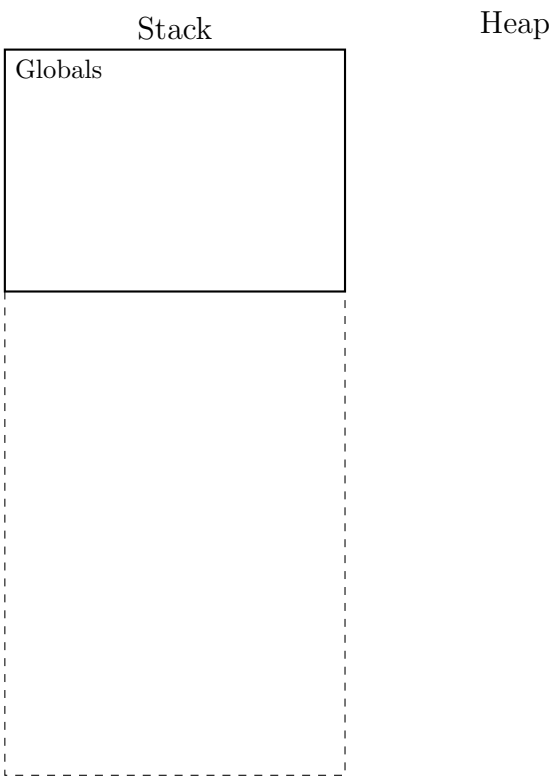
1.1 The University is preparing for graduation and they created a program that allows them to check which students to allow to sit in the seats. In the space below, create a function called `bouncer` that takes in a `Student` object and an array of `Student`s and returns a `boolean`. Imagine that the `Student[]` contains all of the `Student`s enrolled in the University (each `Student` having their own individual properties). The function should meet the following criteria by checking each property individually, and in the following order:

1. If the `Student` is not graduating in 2019, print "Sorry, not yet :(" and return `false`.
2. If the `Student` is not an undergraduate, print "This is not the graduation you are looking for." and return `false`.
3. If the `Student`'s name does not match any in the list of `Student` objects, print "Sorry, you're not on the list :(" and return `false`. To help with this requirement, consider using a loop to iterate through the list and returning `true` if they *are* on the list before notifying them if they are not on the list.

Only one of these messages should ever be printed to the screen.

2. Given the code listing below, draw an environment diagram then answer the questions that follow. In each frame on the stack, remember to include a space for the return address and return value, if any. **NOTE:** In the stack frame for `main`, leave extra room before placing another frame on the stack. You will need space for the `for` loop block in `main` that happens when you return back to `main` from another function call.

```
1  class Bar {
2      n: number = 0;
3  }
4
5  export let main = async () => {
6      let n = 5;
7      let a = [10];
8      let o = new Bar();
9      o.n = 20;
10
11     print(n + "␣" + a[0] + "␣" + o.n);
12
13     fun(n, a, o);
14
15     for (let i = 0; i < a.length; i++) {
16         a[i] = a[i] / 5;
17     }
18
19     print(n + "␣" + a[0] + "␣" + a[1] + "␣" + o.n);
20 };
21
22 let fun = (n: number, a: number[], o: Bar): void=>{
23     n = n * 3;
24     a[0] = a[0] * 5;
25     a[1] = a[0] + 50;
26     o.n = o.n * 2;
27     print(n + "␣" + a[0] + "␣" + a[1] + "␣" + o.n);
28 };
29
30 main();
```



The following questions are about the state of the program after it has finished running. If a question is asked about a name that has not been initialized, then respond with `undefined`.

2.1 From `fun`'s frame, use name resolution to look up the value of the name `n`. What is the value?

2.2 From `main`'s frame, use name resolution to look up the value of the name `n`. What is the value?

2.3 From `fun`'s frame, use name resolution to look up the value of the expression `a[0]`. What is the value?

2.4 From `main`'s frame, use name resolution to look up the value of the expression `a[0]`. What is the value?

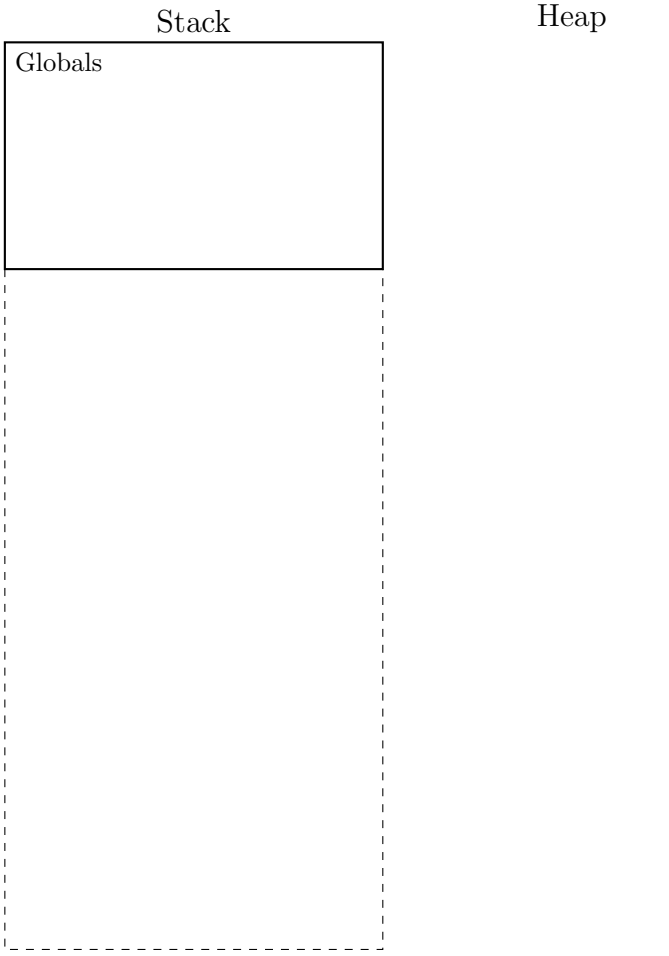
2.5 From `fun`'s frame, use name resolution to look up the value of the expression `o.n`. What is the value?

2.6 From `main`'s frame, use name resolution to look up the value of the expression `o.n`. What is the value?

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

3. Given the code listing below, draw an environment diagram then answer the questions that follow. In each frame on the stack, remember to include a space for the return address and return value, if any.

```
1  class Bear {
2      color: string = "brown";
3      weight: number = 0;
4      speech: string = "roar";
5  }
6
7  export let main = async () => {
8      let grizzly: Bear = new Bear();
9      weighIn(grizzly);
10
11     let polar: Bear;
12     polar = new Bear();
13     polar.color = "white";
14     weighIn(polar);
15
16     let black: Bear = new Bear();
17     black.color = "black";
18     weighIn(black);
19
20     let bears = [polar, black, grizzly];
21     print(bears[bears.length - 2].weight);
22 };
23
24 let weighIn = (bear: Bear): void => {
25     if (bear.color === "brown") {
26         bear.weight = 600;
27     } else if (bear.color === "white") {
28         bear.weight = 990;
29     } else {
30         bear.weight = 180;
31     }
32     print(bear.speech);
33 };
34
35 main();
```



The following questions are about the state of the program after it has finished running. If a question is asked about a name that has not been initialized, then respond with `undefined`.

3.1 From the final `bearApproaching` frame on the stack, what is the value of `bear.color`?

3.2 From `main`'s frame, what are the values of `grizzly.color`, `grizzly.weight`, `grizzly.speech`, respectively?

3.3 On what line is the Bear object that is assigned to the variable `polar` added to the heap?

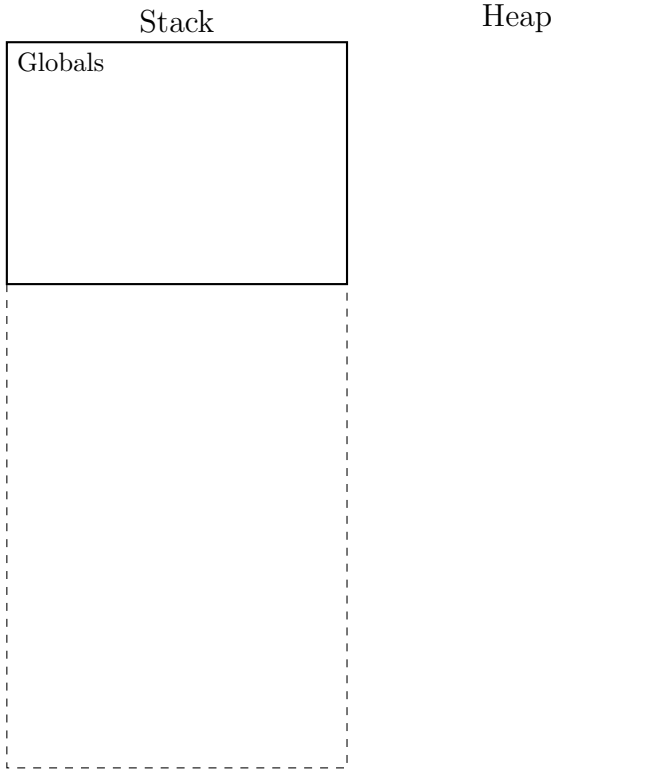
3.4 How many objects and arrays, respectively, are on the heap at the time the program is finished running?

3.5 After the program completes, what is the type of the array on the heap?

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

4. Given the code listing below, draw an environment diagram **paused at the moment line 15** is reached.

```
1  export let main = async () => {
2      let a: string[] = ["a"];
3      let b: string[] = ["b"];
4      swap(a, b);
5      print(a[0] + "␣-␣" + b[0]);
6  };
7
8  let swap = (a: string[], b: string[]): void => {
9      a[0] = "X";
10     b[0] = "Y";
11     let temp = a;
12     a = b;
13     b = temp;
14     print(a[0] + "␣-␣" + b[0]);
15     // Pause Here!
16 };
17
18 main();
```



Questions 4.1-4.6 are about the state of the program at the moment the evaluation is **paused at line 15**. If a question is asked about a name that has not been initialized, then respond with `undefined`.

4.1 From `swap`'s frame, use name resolution to look up the value of the expression `temp[0]`. What is the value?

4.2 From `swap`'s frame, use name resolution to look up the value of the expression `a[0]`. What is the value?

4.3 From `swap`'s frame, use name resolution to look up the value of the expression `b[0]`. What is the value?

4.4 From `main`'s frame, use name resolution to look up the value of the expression `temp[0]`. What is the value?

4.5 From `main`'s frame, use name resolution to look up the value of the expression `a[0]`. What is the value?

4.6 From `main`'s frame, use name resolution to look up the value of the expression `b[0]`. What is the value?

4.7 What is the printed output of this program once it completes, running through the program without the pause on line 15?