COMP110 - Optional WS6 - Name: _____

**1.** The code below defines an `Animal` class which has a name, food amount, and size property.

```
1  class Animal {
2      name: string;
3      foodAmount: number;
4      size: string;
5
6      constructor(n: string, f: number, s: string) {
7          this.name = n;
8          this.foodAmount = f;
9          this.size = s;
10     }
11 }
12
13 let main = async () => {
14     let zoo: Animal[] = [
15         new Animal("Giraffe", 15, "large"),
16         new Animal("Beaver", 6, "small"),
17         new Animal("Elephant", 27, "large"),
18         new Animal("Flamingo", 5, "medium"),
19         new Animal("Ferret", 3, "small"),
20         new Animal("Otter", 4, "small"),
21         new Animal("Beetle", 1, "tiny")
22     ];
23     // where you would place your function calls
24 };
```

**1.1** Write a predicate function named `animalsBySize` which will return true if the `size` property of the `Animal` parameter is `"small"` and false otherwise.

**1.2** Write a transform function named `toFoodAmount` which will return only the `foodAmount` property of the `Animal` object passed in.

**1.3** Write a reducer function named `total` which will return the sum of two number parameters.

**1.4** Using your new functions and the built-in array `filter`, `map` and `reduce` methods, write one or more function calls that would be placed at line 23 to find the total amount of food that the zookeeper will need to bring in order to feed the `"small"` animals; store this answer in a variable called `foodNeeded`. You may create new variables to store data if you would like.

**2.** Given the code to the left, answer the questions that follow.

```
1  class ArrHolder {
2      arr1: number[];
3      arr2: number[];
4
5      constructor(a1: number[], a2: number[]) {
6          this.arr1 = a1;
7          this.arr2 = a2;
8      }
9
10     together() {
11         for (let i = 0; i < this.arr2.length; i++) {
12             let len = this.arr1.length;
13             this.arr1[len] = this.arr2[i];
14         }
15         return this.arr1; // A
16     }
17 }
18
19 let arrMixer = (a: number[], b: ArrHolder) => {
20     for (let i = 0; i + 1 < a.length; i++) {
21         a[i] = a[i + 1];
22     }
23     b.arr1 = [4, 5];
24     return a;
25 };
26 let main = async () => {
27     let favs = new ArrHolder([1, 4, 2, 3], [3, 6]);
28     print(arrMixer(favs.together(), favs)); // B
29     print(favs.arr1); // C
30     print(favs.arr2); // D
31 };
32
33 main();
```

**2.1** What is the value of `favs.arr1` when comment A is reached? (line 14)

**2.2** What is the value of `favs.arr2` when comment A is reached?

**2.3** What is printed at comment B?

**2.4** What is printed at comment C?

**2.5** What is printed at comment D?

**3.** Given an array of numbers named `arr`, use a function literal and the built in `filter` method to filter out all of the numbers in the array that are not prime, and store this result in a variable named `primes`. Note: By definition, prime numbers must be greater than 1.

**4.** Given the following new House object created below, write out its class definition including the constructor. (Hint: Name your properties with respect to features of a house.)

```
let blueHouse:  House = new House("blue", 2, 4);
```

**5.** Given the code to the left, answer the questions that follow. Functions you write would be placed below line 28.

```
1  class Vehicle {
2      color: string;
3      doors: number;
4      type: string;
5      price: number;
6      description: string;
7
8      constructor(color: string, doors: number, type:
           string, price: number) {
9          this.color = color;
10         this.doors = doors;
11         this.type = type;
12         this.price = price;
13         this.description = doors + "␣door␣" + color
               + "␣" + type + "␣for␣$" + price;
14     }
15 }
16
17 class Advertisement {
18     vehicle: Vehicle;
19     promocode: string;
20     output: string;
21
22     constructor(v: Vehicle, promo: string) {
23         this.vehicle = v;
24         this.promocode = promo;
25         this.output = "New␣" + v.description + "␣on␣
               sale␣now!␣Use␣promo␣code␣" + promo + "␣
               for␣5%␣off!";
26     }
27 }
```

**5.1** Write a function named `filterAdByVehicleColor` that will take in an `Advertisement` and return true if the `Advertisement's Vehicle` property's `color` is `"blue"`.

**5.2** Write a function named `mapToVehicle` that will take in an `Advertisement` and return its `Vehicle` property.

**5.3** Write a function named `reduceByPrice` that takes in two `Vehicles` and returns the `Vehicle` with the lowest price, or the second if they are equal.

**5.4** Write a function named `findDreamCar` that will take in an array of `Advertisements` and return a `Vehicle`. You should use the `filterAdByVehicleColor`, `mapToVehicle`, and `reduceByPrice` functions and their respective `filter`, `map`, and `reduce` functions. Your code should output the `Vehicle` that is blue and the cheapest, and should utilize all of the functions you created prior, as well as the built-in array filter, map, and reduce methods.

**6.** Put a check by the functions that correctly use the TheOffice interface.

```
interface TheOffice<J, D> {
    (a: J, b: D): J;
}
```

**6.1** .

```
let jim: TheOffice<boolean, boolean> = (b: boolean,
     n: number): boolean => {
    return "jim";
};
```

**6.2** .

```
let dwight: TheOffice<boolean, number> = (b:
    boolean, n: number): boolean => {
    return true;
};
```

**6.3** .

```
let pam: TheOffice<number, number> = (b: boolean, n
    : number): boolean => {
    return "1";
};
```

**6.4** .

```
let angela: TheOffice<string, number> = (b: string,
     n: number): string => {
    return "false";
};
```