Question 1

1. No error
2. No error
3. Static semantic error
4. Static semantic error
5. Static semantic error
6. Not a compile time error
7. No error
8. No error
9. Static semantic error
10. Syntax error
11. Syntax error
12. Syntax error
13. Not a compile time error

Question 2

- The code in javaScript returns a Syntax error since x is declared twice.
- The code in Rust compiles without an error

Question 3

- In Java, *private* is a class modifier that restricts access to that method so it is only visible inside the same class.
- In Ruby, private prevents a method from being called with an explicit receiver, but does not restrict the subclass access.

Question 4

if your language does not require your parameters to be in a certain order, then results of the function may vary depending on the function.

Question 5

The Carlos language handles recursive structures similarly to other languages by creating a struct/method to handle the recursion. In the structure, no field of the structs is allowed to have the type of the struct itself, as that would lead to an infinitely-sized structure. Therefore, recursion must be done using optionals, arrays, or functions that use the type itself, However, Calros contains the repeat method used specifically for integers which can be used to recursively call a portion of code until an integer is evaluated equal to another integer. (*e* is equal to *t*) Another restriction on recursion in the language includes no shadowing. This means that all variables inside of the recursive method must be their own unique variables and they cannot have the same names of variables outside of the loop.

Question 6

Python

```
def min_array(arr):
    """
    Returns the minimum value of a list of floats using tail recursion.
    No loops. No external/global state.
    """

    def min_tail(lst, current_min):
        if not lst:  # Base case: no elements left
            return current_min

        new_min = lst[0] if lst[0] < current_min else current_min
        return min_tail(lst[1:], new_min)

    return min_tail(arr, float('inf'))
```

C

```
#include <stdio.h>
#include <math.h>
```

```
float min_tail(const float *arr, int n, float current_min) {
    if (n == 0) {
        return current_min;
    }
    float new_min = (arr[0] < current_min) ? arr[0] : current_min;
    return min_tail(arr + 1, n - 1, new_min);
}

float min_array(const float *arr, int n) {
    return min_tail(arr, n, INFINITY);
}
```

JavaScript

```
function minArray(arr) {
  function minTail(index, currentMin) {
    if (index >= arr.length) {
      return currentMin;
    }
    const newMin = arr[index] < currentMin ? arr[index] : currentMin;
    return minTail(index + 1, newMin); // Tail call
  }

  return minTail(0, Infinity);
}

// Example usage:
// console.log(minArray([3.14, 2.71, 6.28, -1.0, 0.0])); // => -1
```

Rust

```
fn find_min(arr: &[f64]) -> f64 {
    fn find_min_tr(slice: &[f64], current_min: f64) -> f64 {
        if slice.is_empty() {
            current_min
```

```
        } else {
            let new_min = if slice[0] < current_min {
                slice[0]
            } else {
                current_min
            };
            find_min_tr(&slice[1..], new_min)  // Tail call
        }
    }

    find_min_tr(arr, f64::INFINITY)
}
```

Question 7

The function when called returns a TypeError. This error occurs because the setTimeout function is not being called correctly. The way it is being called in the code, “I” does not increment every second which causes the error. Here is a correct sample:

```
function countDownFromTen() {
    function update(i) {
      document.getElementById("t").innerHTML = i;
      if (i > 0) {

        setTimeout(function () {
          update(i - 1);
        }, 1000);
      }
    }
    update(10);
  }
```

Question 8

These are the errors that occur:

'import ... =' can only be used in TypeScript files
Unknown keyword or identifier. Did you mean 'finally'?
Type parameter declarations can only be used in TypeScript files
'(' expected
';' expected
Unexpected keyword or identifier
Expression expected
Declaration or statement expected
unexpected keyword or identifier
';' expected
Declaration or statement expected
';' expected