1) MEDIUM

def double\_numbers(numbers):

for i in range(len(numbers)):

numbers[i] = numbers[i] \* 2

return numbers

my\_list = [1, 2, 3, 4, 5]

doubled\_list = double\_numbers(my\_list)

print(doubled\_list)

//double\_numbers modifies the original list

2) MEDIUM

function calculateAverage(nums) {

var sum;

for (var i = 0; i < nums.length; i++) {

sum += nums[i];

}

var avg = sum / nums.length;

return avg;

}

var nums = [1, 2, 3, 4, 5];

var avg = calculateAverage(nums);

console.log(avg);

//unitialized variable

3) HARD

public class BugExample {

public static void main(String[] args) {

int x = 5;

int y = 10;

int z = addNumbers(x, y);

System.out.println("The sum of x and y is: " + z);

}

public static int addNumbers(int a, int b) {

int sum = a + b;

return sum;

}

}

//what if x + y overflows the int data type?

4) HARD

public function show($id)

{

$user = User::find($id);

if ($user) {

return view('user.show', compact('user'));

} else {

return view('error.404');

}

}

//returns 200 status code when user is not found

5) EASY

#include <iostream>

int main() {

int x = 5;

int y = 10;

int sum = x + y;

std::cout << "The sum of x and y is: " << sum << std::endl;

int product = 1;

for (int i = 0; i <= 10; i++) {

product \*= i;

}

std::cout << "The product of the first 10 integers is: " << product << std::endl;

int z = sum \* x + y;

std::cout << "The result of the calculation is: " << z << std::endl;

return 0;

}

//Loop should start from 1 not 0

6) EASY

#include <stdio.h>

int main() {

int arr[5];

arr[0] = 1;

arr[1] = 2;

arr[2] = 3;

arr[3] = 4;

for (int i = 0; i < 5; i++) {

printf("Value at index %d is: %d\n", i, arr[i]);

}

return 0;

}

// Loop should stop at 4 not 5

7) EASY

myNumber = 1

if(myNumber < 10):

addOne(myNumber);

function addOne(myNumber):

myNumber += 1

//no semicolons in python

8) HARD

.container {

width: 100%;

height: 100%;

position: relative;

}

.item {

position: absolute;

top: 50%;

left: 50%;

transform: translate(-50%, -50%);

width: 200px;

height: 200px;

background-color: #ccc;

}

.item:hover {

background-color: #f00;

width: 300px;

height: 300px;

}

//when hovering, the item becomes bigger than its container so height should use vh

//The bug in this code is that the hover effect causes the item element to expand beyond the //boundaries of the container, which is not the intended behavior. This is because the container //element has a height of 100%, which is relative to the height of its parent element, but the parent //element has no specified height, so it defaults to the height of its content. When the item element //expands, it exceeds the height of the container and spills out of it.

9) EASY

.item:hover {

backgroundColor: #f00;

width: 300px;

height: 300px;

}

//Should be background-color

10) MEDIUM

def compute\_average(num\_list):

"""

Compute the average of a list of numbers.

"""

total = 0

num\_items = 0

for num in num\_list:

total += num

num\_items += 1

average = total / num\_items

return average

//should check for empty list because we would have division by zero

11) MEDIUM

#include <stdio.h>

int main() {

double x = 25.0;

double result = sqrt(x);

printf("The square root of %lf is %lf\n", x, result);

return 0;

}

//missing import statement

12) EASY

public class Test {

public static void main(String[] args) {

int a = 5;

int b = 10;

int sum = add(a, b); // we forgot to pass the third parameter

System.out.println("The sum is: " + sum);

}

public static int add(int x, int y, int z) {

return x + y + z;

}

}

//add should take 3 parameters

13) EASY

def calculate\_average(numbers)

total = 0

for num in numbers:

total += num

average = total / len(numbers)

return average

nums = [3, 7, 2, 8, 4]

avg = calculate\_average(nums)

print("The average is:", avg)

//missing colon

14) EASY

#include <stdio.h>

void print\_fibonacci(int num) {

printf("Fibonacci series up to %d terms:\n", num);

calculate\_fibonacci(num);

}

void calculate\_fibonacci(int num) {

static int first\_term = 0, second\_term = 1, next\_term, count = 0;

if (count < num) {

printf("%d ", first\_term);

next\_term = first\_term + second\_term;

first\_term = second\_term;

second\_term = next\_term;

count++;

calculate\_fibonacci(num);

}

}

int main() {

int num\_terms;

printf("Enter the number of terms: ");

scanf("%d", &num\_terms);

print\_fibonacci(num\_terms);

return 0;

}

//wrong order of functions

15) MEDIUM

<?php

$string = "Hello, world!";

$substring = substr($string, 7, 12);

echo $substring;

?>

//substr takes the length of the substr as its third argument, not the ending index