Midterm Exam 1

Fall 2020 COP3223

Short Answers: 8 points each

Your answers to these questions should each be one paragraph or less.

Reflections

1. Briefly (in one paragraph or less) describe all the steps involved in creating a program with the tools we've used.

Firstly, we open our projects directory and create a new folder for our new program. We then open VScode in the new directory created for the program. We create a file such as test.c. .c meaning a c file. Then we must add #include <stdio.h> so we can print to the terminal, and we then create an int main(void) function that has a return 0. Without a main function, gcc will not be able to find a main function to initialize the code at. From there you can code your program and include other libraries necessary.

2. Why do we indent code?

we indent code for easy readability, and the ability to tell code blocks from one another. Indenting a for loop will allow us easily to determine what code is exactly within that loop. Although brackets help with seeing what is doing what, the indendating really allows the code to stick out in that portion of the program.

3. Why do we comment code?

We comment code to explain what it is our code is doing. In a real world development environment, which I have experienced, the most common comments were at the beginning of a file, simply stating who created the file and what it does. With good enough naming conventions, and clean code, it should be easy enough to read, with experience, without comments, however, for this class purpose, commenting functions, loops, etc, for our programs will help other people viewing the code understand what the code will be doing. We simply add comments to give explanations.

4. Why do we use functions?

We use functions in our programs to separate our code out into portions that make the program more easily understandable. Having everything in 1 function can lead to confusion and difficulty in understanding the program. Separating out the code into functions with specific tasks, helps the flow of reading and understanding the program. Functions can also be used to do things such as math outside of the main function and return the value the math provides. Object oriented code is even more easily readable because we are able to have neat, clean code with functions and variables in separate files with a simplistic main file :)

Details

5. Explain the difference between **int** and **float**.\

An integer is a whole number ex: 1,2,3,4 while a float is a floating decimal, such as ex: 1.5, 2.8, 3.9. Integers may not hold decimal values.

6. Explain the difference between **float** and **double**.

The difference between a float and double is within their precision. A double has a 64 bit precision while a float has 32 bit precision. Making a double twice as precise as a float :)

7. `j = 5 * i;` contains both a statement and an expression. Explain.

This code contains a statement because we are setting j equal to something. Setting a variable to something is considered a statement. It also includes an expression, because 5*i or multiplication in general is an expression. More specifically, 5*i contains 2 operands (5 and i) and an operator(*), making it an expression.

8. Explain the difference between the **while** and the **do-while** loops.

The difference between a do-while loop and a while loop, is that the do-while loop executes the code once before checking the condition of the while statement. While loops will check the condition before executing any code.

Problems: 9 points each

Each of these very short programs has something wrong with it. Explain what. Your answers should each be one sentence or less.

Program 1

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

int main(void) {
    double x = 9.0;

    y = sqrt(x);
    printf("SQRT(%lf) is %lf.\n", x, y);

    return 0;
}
```

Y is missing an initialization of it being a double. There needs to be: double y = sqrt(x); Without the double, y will not know what variable type it is. Depending on the compiler, y may default to an int, or error out.

Program 2

```
#include <math.h>

int main(void) {
    double x = 3.0;
    double y = 4.0;

    printf("The sum is %lf.\n", x + y);

    return 0;
}
```

This program is missing the inclusion of the <stdio.h> library, meaning it will not be able to print to the terminal with printf. It needs to have: #include <stdio.h> to work correctly.

ALL these programs Mr. Gerber… JavaScript style brackets… shameful.

Program 3

```
#include <stdio.h>

double square(x)
{
    return (x * x);
}

int main(void) {
    double x = 3.2;
    double y;

    y = square(x);
    printf("%lf squared: %lf.\n", x, y);

    return 0;
}
```

This program is missing the parameter type within the square(x). This function should be declared with double for x such as: double square(double x). This will cause an error, or sometimes just default x to an int depending on the compiler.

Program 4

```
#include <stdio.h>

int main(void) {
    printf(Hello, world!\n);
    return 0;
}
```

This program is simply missing quotation marks around the text printf is attempting to print. Without the quotation marks, this program will error out and not know what to print.

ALL these programs… where are your comments Mr. Gerber??? Very interesting…