**Programming Project Report**

Morgan Maness

1/24/18

Task 1: Road Trip

The goal of the road trip assignment was to calculate how far someone could drive given how much money they had, the price of gas, how much gas they had in their car right now, and their car’s mile per gallon. Those were the inputs and the output was how far they could drive. My program prompts the user for each variable one at a time, and while I was writing my code I printed out each input as soon as they entered it in order to ensure that I was doing everything correctly, but after it was written I deleted those output lines in order to make my code more user friendly and take up as little space as possible. For my error handling, I checked each input to ensure that it as a positive number, since you cannot have negative gas in your car. If the number was a negative, or zero for miles per gallon and price of gas, my program would prompt the user to input a valid variable number.

I broke up my code into three parts. In the first part I created the variables. Next, I prompted the user to input values for my variables, and lastly, I calculated the total distance the user can drive with the variables give. I used floats for my variables in order to make my output as accurate as possible, and when I outputted my data, I used setpercision to four so that it would not output some massive number. The algorithm that I used to calculate the total distance was ((total money/price for gas) + current gas) \* miles per gallon. The pros of calculating it like this is that I don’t need to add any extra variables to hold my calculations along the way, but the cons of this is that if my calculations were messed up in any way, it would be very hard for me to pin point the exact place where I messed up without dissecting my equation.

After I set up the three sections of my code, I just started from the top and took each section one at a time. Because the code is relatively small, I was able to compile and do a practice run every time I finished a section. Once I had my code working perfectly, I went back and added my while statements to not take negative numbers. The only sample code that I used was the code that my compiler (Xcode) started me with, which is the basic skeleton for a hello world code. I simply deleted the hello world output and started from scratch there. Overall, it took me around twenty minutes to create my code, with probably the most time spent on creating the equation for my output. The only reason that took as long as it did was because it took me a while to decide how I wanted to calculate the distance that the user could travel.

I tested my program at every step of creation. Once I made sure that one section of my code worked, I went on to the next. After everything worked the way I needed it to, I started testing the entire code in every way I could think of. The normal inputs that I used were the ones that the instructions gave me, plus some random numbers I came up with on my own. The special cases that I tested were negatives and zeroes, which is when I decided to add the while statements into my code. After I threw the while statements in, my code worked exactly as expected.

Sample input and output for my code

**Available gas money:** 40

**Price of gas:** 0

**Invalid input, try again:** 2.25

**Miles per gallon of your car:** 28.6

**Gas in your car right now:** -15

**Invalid input, try again:** 5

**You can drive 651.4 miles south.**

Task 2: Physics

The goal of the physics programming assignment was to calculate the Lorentz factor, which describes how time is affected while going extreme velocities in space. The programs input is the velocity, and its output is how the time that you feel is relative to one year on Earth. Error handling that I encountered was possibly converting the velocity input to kilometers per hour. Because the velocity has to match the constant for the Lorenz factor, I gave the user the chance to convert a velocity in miles per hour to kilometers per second.

I broke up my code into four sections. The first section was creating the variables. The second was getting the velocity from the user. Then I converted the results as needed, and calculated the result. My last section was to output the result. I used floats for my variables, and an integer for if the user needed to convert. For my algorithm, I broke up the calculation into parts, in order to decrease the possibility of me messing up somewhere. The pros of my choice were that I knew the algorithm was perfect, and the cons of my choice was that the code is a lot longer than it could be and somewhat cumbersome.

Much like the first section of my code, I took my code one step at a time. Since this part of the code was in the same code as the road trip code, I had the skeleton code already set up. Other than that, I started the rest of my code from scratch. My timeline was going by sections, and the longest part was the calculations. Because it was impossible for me to just type in the algorithm as I saw it, I had to take it step by step and that ended up taking a fair amount of time to get right.

In order to test my program, I first used the sample information from the instructions, then I tested that sample data in miles per hour to ensure my conversion was correct. Then I tested random numbers to ensure that everything was working properly. The special cases that I tested were my inputs that were in miles per hour. Everything worked exactly as expected once I entered my algorithm in correctly.

Sample input and output for my code

**Enter velocity: 50000**

**If your velocity is in miles per hour, (instead of Kilometers per second) type 1. 0**

**1.014 years will pass on Earth for every one year the astronaut continues at this speed.**

Overall, the results of this assignment were a success. I was able to take inputs from the user and use them to calculate the desired end results. Next time I would work to code the program cleaner. I broke apart a lot of my code in order to ensure that all my algorithms worked correctly. For the entire project, including the write up, the entire project took me around an hour and a half to complete.