Created by: John Villalvazo and Uriel Montes 4/11/2019

Psuedocode/Notes for CSE31 Project 2

1. Overall plan/needs

* Need to complete the 3 main functions so that recursively we can find out how bars of candy can a person get with a set amount of initial money
* Only "input" that will be going into this program is the persons limit on money (ex. $12) every other input within the program is devised from this initial amount because of it causes for different amounts of physical candy bars as well as different amounts of wrappers that allow the user to get free bars.
* Provided Code (from a initial view) seems to run in succession. Meaning that if starting with MAIN, technically can be used as a gauge to see what next needs to be dealt with/coded next.
* Code with (meaning :along with) the comments, seeing how these comments are giving a detailed view as to what is expected next, part by part can be pieced together to get a general overview of what needs to be done
* Generate C code from MIPS code to have a physical guide to follow while completing this MIPS project.

Pseudocode in low-level (non-language specific):

Will Need:

* String to hold opening statement
* Variable to hold user inputted money amount
* Variable to hold cost of one(1) BobCat Bar
* Variable to hold amount of wrappers to trade in for 1 free candy bar
* String to hold max amount of Bobcat Bars user will receive after combining all outputs
* Function to find out the number of candy bars your money can buy

Function to find out how many candy bars you can get with the number of wrappers

C Code (Sentence format)

Begin with a main function that holds all the inputted data from the user into variables. For each code block , print a statement that tells the user what type of information needs to be provided. Have that data be saved into a variable to be held for later use in the helper functions. Rinse and repeat for all required statements. After getting all required inputs, find the initial amount of bars that can be purchased with by diving the price by the enter amount of money a person has. Print that statement so that the user knows how many initially they can afford. After that, call upon the maxBars helper function to assist with recursively finding out how many bars in total can be received.

Within the maxBars helper function, the arguments of the price of each bar, amount needed for an exchange and the total amount of money a person has will assist with finding out how many bars a person will receive. We again find out the initial amount a person can afford and place it into a variable. Using that information, along with declaring any other variables to assist the user, we must use another helper function to find out recursively how many more bars can be given to the user by parsing through the values of the initial and the number required for an exchange.

The helper function newBars will allow the program to find how many more bars can the user receive by exchanging the bars wrappers for another candy bar. There should be some sort of check to find out if there are no more wrappers, if there are still a way for there to receive more candy bars in exchange for wrappers then continue to call this function. This value of total free (exchanged) bars should be maintained within this scope until this check has completed in which then it can be returned to the previous function (maxBars).

Once all these calculations are completed, the value of total free bars should then be added up to get the overall amount of candy bars. The should be relayed to the user so that they know how many bars are given to the user for the initial amount of money they have.

View attached c code