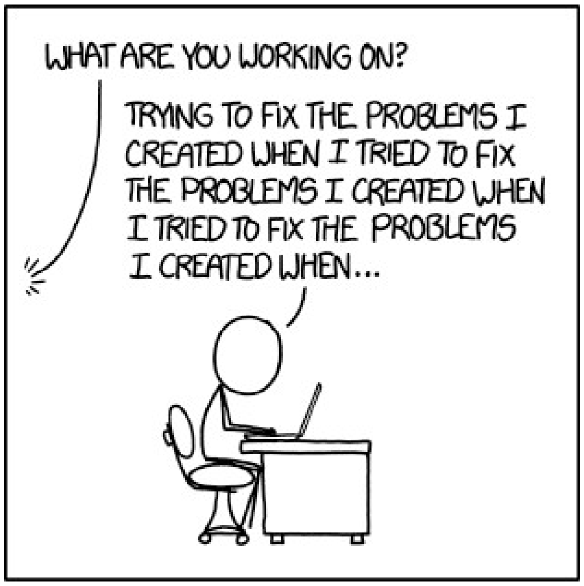
CS 218 – MIPS Assignment #5

Purpose:	Become familiar with the MIPS Instruction Set, and the MIPS procedure calling convention, and basic recursion.
Points:	100

Assignment

Write a MIPS assembly language program to count the number of ways that change can be made given the our standard US monetary denominations (cent, nickel, dime, quarter, fifty cent, dollar, and five dollar note). For example, for 15 cents, there are 6 different ways to make change.

	Method
1	Dime (1), Nickel (1)
2	Nickel (3)
3	Dime (1), Penny (5)
4	Nickel (2), Penny (5)
5	Nickel (1), Penny (10)
6	Penny (15)


WHAT ARE YOU WORKING ON?
TRYING TO FIX THE PROBLEMS I CREATED WHEN I TRIED TO FIX THE PROBLEMS I CREATED WHEN I TRIED TO FIX THE PROBLEMS I CREATED WHEN...

Source: https://xkcd.com/1739/

Using the provided main, write the following assembly language procedures:

- Write a MIPS boolean function, ***readInitAmt(min, max, errLimit, &initAmt)***, to prompt (provided) for and read the initial amount value, in cents, from the user. The amount must be between ***min*** and ***max*** (inclusive). If valid input is provided, the function should return TRUE. If invalid input is provided, display an error message (provided) and re-prompt up to ***errLimit*** times. If a ***errLimit+1*** errors are made, the function should display the error message and return false.
- Write a recursive MIPS integer function, ***makeChange(coins, denominationCount, currAmount)***, to determine the number of ways to make change for the passed monetary value. Must be recursive for credit. The recursive relation is as follows:

$$mkChg(c[], cCnt, amt) = \begin{cases} 1 & \text{if } amt = 0 \\ 0 & \text{if } amt < 0 \\ 0 & \text{if } cCnt \le 0 \\ mkChg(c[], cCnt-1, amt) + \\ \quad mkChg(c[], cCnt, amt - c[cCnt-1]) & \text{otherwise} \end{cases}$$

- Write a MIPS void function, ***showWays(waysCount)***, to print the formatted results (i.e., "For a value of $*xx.yy* , there are *zz* ways to make change."). You are ***not*** required to list them. Refer to the example output for formatting.

- Write a MIPS value returning function, ***continue()***, to prompt the user ("Try another amount (y/n)?") if they want to enter another value or exit the program. If invalid input is provided, display an error message (provided) and re-prompt. *Note*, you only need to check for a lower case response.

Refer to the example for formatting. You may define additional variables as required.

Submission

- All source files must assemble and execute with QtSpim/SPIM MIPS simulator.
- Submit source file
  - Submit a copy of the program source file via the on-line submission
- Once you submit, the system will score the project and provide feedback.
  - If you do not get full score, you can (and should) correct and resubmit.
  - You can re-submit an unlimited number of times before the due date/time (at a maximum rate of 5 submissions per hour).
- Late submissions will be accepted for a period of 24 hours after the due date/time for any given assignment. Late submissions will be subject to a ~2% reduction in points per an hour late. If you submit 1 minute - 1 hour late -2%, 1-2 hours late -4%, … , 23-24 hours late -50%. This means after 24 hours late submissions will receive an automatic 0.

Program Header Block

All source files must include your name, section number, assignment, NSHE number, and program description. The required format is as follows:

```
#  Name: <your name>
#  NSHE ID: <your id>
#  Section: <section>
#  Assignment: <assignment number>
#  Description: <short description of program goes here>
```

Failure to include your name in this format will result in a reduction of points.

Scoring Rubric

Scoring will include functionality, code quality, and documentation. Below is a summary of the scoring rubric for this assignment.

Criteria	Weight	Summary
Assemble	-	Failure to assemble will result in a score of 0.
Program Header	3%	Must include header block in the required format (see above).
General Comments	7%	Must include an appropriate level of program documentation.
Program Functionality (and on-time)	90%	Program must meet the functional requirements as outlined in the assignment. Must be submitted on time for full score.

Example Output
The following are some examples of correct output.

Example 1:

```
**********************************************
MIPS Assignment #5
Ways to Make Change Program
  Enter Amount (1 - 500): 105


For an amount of $1.05, there are 337 ways to make change.



Try another amount (y/n)? q
Error, must answer with (y/n).
Try another amount (y/n)? 4
Error, must answer with (y/n).
Try another amount (y/n)? y

  Enter Amount (1 - 500): 225


For an amount of $2.25, there are 4166 ways to make change.



Try another amount (y/n)? y

  Enter Amount (1 - 500): 500


For an amount of $5.00, there are 98412 ways to make change.



Try another amount (y/n)? n


You have reached recursive nirvana.
Program Terminated.
```

Example 2:

```
**********************************************
MIPS Assignment #5
Ways to Make Change Program
  Enter Amount (1 - 500): 15


For an amount of $0.15, there are 6 ways to make change.



Try another amount (y/n)? y

  Enter Amount (1 - 500): 1000

Error, amount out of range.
Please re-enter data.
  Enter Amount (1 - 500): -3

Error, amount out of range.
Please re-enter data.
  Enter Amount (1 - 500): 3000

Error, amount out of range.
Please re-enter data.
  Enter Amount (1 - 500): 5000

Sorry your having problems.
Please try again later.

You have reached recursive nirvana.
Program Terminated.
```

Example 3:

```
**********************************************
MIPS Assignment #5
Ways to Make Change Program
  Enter Amount (1 - 500): 105


For an amount of $1.05, there are 337 ways to make change.



Try another amount (y/n)? y

  Enter Amount (1 - 500): 203


For an amount of $2.03, there are 2728 ways to make change.



Try another amount (y/n)? y

  Enter Amount (1 - 500): 78


For an amount of $0.78, there are 134 ways to make change.



Try another amount (y/n)? y

  Enter Amount (1 - 500): 15


For an amount of $0.15, there are 6 ways to make change.



Try another amount (y/n)? y

  Enter Amount (1 - 500): n

Error, amount out of range.
Please re-enter data.
  Enter Amount (1 - 500): 107


For an amount of $1.07, there are 337 ways to make change.



Try another amount (y/n)? n


You have reached recursive nirvana.
Program Terminated.
```