TopGear Assignments

Google Go Programming – L1

Set - 1

Estimated Effort: 1 PD
Total Points: 50

Date: 27-Jan-2020
Last edited: 25-May-2021

Required VDI: TG-DOTNET

This set of assignments requires use of the following:

- String operations
- Command-line arguments
- Functions

Code submitted should meet the following guidelines: -

- Code filename should be as specified in each exercise
- Data validation & error handling has to be part of the solution
- Code should not be restricted to work only for those examples given if any.
- Input values should not be hard coded in programs.
- Program should not produce output other than in expected
- Individual **.go** files have to be uploaded. Not as ZIP or .doc files.

Exercise-1 [filename: ex1.go]

Implement the function **num2words()** which takes a number and returns the number in words as a string.

For example, **num2words(123)** should return the string “one two three”.

Test correctness of the function in main(), as per the examples given below.

Command	Output expected
go run ex1.go 123	one two three
go run ex1.go 7540 468	seven five four zero four six eight
go run ex1.go 35a	Not a number

Exercise-2 [filename: ex2.go]

Implement the function **maxPower()** that takes two parameters **M** and **N** and returns the smallest integer **K** such that $M <= N^K$

maxPower (80000, 5) should return 8

maxPower (30000, 9) should return 5

Test correctness of the function in main(), as per the examples given below.

Command	Output expected
go run ex2.go 80000 5	8
go run ex2.go 30000 9	5

Exercise-3 [filename: ex3.go]

Implement the function **words2num() which** takes a string and returns the number represented by the string.

> For example, words2num(“one two three”) should return 123.
> Function argument can be in lowercase/uppercase/mixedcase.

Test correctness of the function in main(), which takes a string as commandline arguments, calls words2num() & prints the returned value.

If any of the word does not represent a digit, program should print an error messages indicating invalid input.

Command	Output expected
go run ex3.go "one two three"	123
go run ex3.go "Four FIVE nine zero"	4590
go run ex3.go "Four FIVR nine zeero"	Invalid input

Exercise-4 [filename: ex4.go]

Implement the function **num2hex()** which takes an unsigned integer and returns its hexa equivalent as a string. Function should take an extra argument which specifies whether the hexa digits should be in lower case (default) or upper case.

Using the function num2hex() implement main which takes an integer value and prints its hexa equivalent. If the number precedes with –u option, then the hexa value should be printed in uppercase. Any other option other than –u should be considered invalid.

Command	Output expected
go run ex4.go 677	2a5
go run ex4.go -u 107471	1A3CF
go run ex4.go -L 512	Invalid option
Go run ex5.go 32a	Invalid number

Exercise-5 [filename: ex5.go]

Write a program that takes 2 complex numbers and the operation to be performed on them as command line arguments, and prints the result as a complex number. Commandline arguments to the program are in the following format

complexnum1 *binaryop* complexnum2

If one of the arguments is in complex form and the other is either in integer or float form, assume its imaginary part is zero (0).

Note: Assume there is no embedded space in complex number argument.

Command	Output expected
go run ex5.go 3-4i + 7+2i	10-2i
go run ex5.go -15i - 3-4i	-3-11i
go run ex5.go 3-5i * 12	36-60i

Note: to run the command in Linux, * should be prefixed with \

- - - X - - -