VG101 — Introduction to Computer and Programming

Homework 4

Manuel — UM-JI (Fall 2020)

- MATLAB: write each exercise in a different file
- C/C++: use the provided assignment template
- Include simple comments in the code
- If applicable, split the code over several functions
- Extensively test your code and impove it
- Write a single README file per homework
- Archive the files (`*.{zip|tar}`) and upload on Canvas

*Exercises preceded by a * are mandatory. Any student not completing all of them, or submitting a work that cannot compile or be interpreted will automatically be deducted 1 mark on the final course grade.*

JOJ Online Judge

Exercises 1 to 3 can be tested on JOJ Online Judge.

Important reminders regarding the Online Judge (OJ):

- For each exercise save all the files, without any folder structure, into a `.tar` archive;
- Strictly stick to the input and output formats provided in the specifications;
- The OJ only checks the correctness of the code not its quality;
- For feedbacks on the quality, submit the code as part of the assignment and include the OJ score as well as the failed cases in the README file;

* **Ex. 1** — *Structure, basic programming*

A complex number is composed of a real part and of an imaginary part. Create a structure to represent complex numbers. Write two functions to calculate the product and the sum of two complex numbers. A third function should ask the user to input two complex numbers together with an operation, and then display the result.

Specifications.

- Input format: two complex numbers and an operator on one line (e.g. `1+2i 3-4i +`)
- Output format: one complex number on one line (e.g. `4-2i`)
- Note: no space is allowed in a complex number; expected format is `sa±bi`, where *s* is − or nothing and *a* and *b* are two numbers.
- Filename: `ex1.c`

* **Ex. 2** — *Recursion, basic arithmetic*

Given two integer:

1. Write a recursive algorithm to calculate their GCD.
2. Translate the algorithm into C.

Specifications.

- Input: a space separated pair of numbers *a b* on one line;
- Output: a single number on one line;
- Filename: `ex2.c` (the `main()` function should only handle the I/O)

Ex. 3 — *Use of the math library, conditional statements*

Write a C function to find the roots of a quadratic equation whose coefficients are input by the user.

Specifications.

- Input: a space separated triple *a b c*, standing for $ax^2 + bx + c$, on one line;
- Output: one root per line with a precision of five decimals;
- Filename: `ex3.c` (the `main()` function should only handle the I/O)

Ex. 4 — *Conditional statements*

1. Write a C function which prompts the user for a functionality and dispatches the work to the corresponding function from question 2 or 3.
2. Write a C function to check whether the letter input by the user is a vowel or a consonant.
3. Write a C function which prompts the user for a number and a character. The character can only be 'b' for bit of 'B' for byte. The function should print all the common data types whose size are equal to the number input by the user. The 'b' and 'B' will determine whether the input number is to be considered as bits or bytes.

Specifications.

- Question 1:
  - Display the list of functionalities to the user
  - Read the user's choice as a digit or a character
- Question 3:
  - At least all the four basic data types should be considered
  - Output exactly one datatype per line

* **Ex. 5 —** *Pre-processing, program structure, data size*

Copy the following code and split it into 5 files: `main.c` (or `ex5.c` when using the C template), `sum.c`, `prod.c`, `quorem.c`, and `exp.c`. Create their corresponding .h files. Add the appropriate `#include` directives. Use the `#ifdef or #ifndef` macros such that it is possible to only compile one operation at a time (for instance the program could only return the quotient and remainder but not the product, the exponent and the sum). Discuss the size of the input/output for the exponent function.

math.c

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

```c
int sum (int a, int b);
int prod (int a, int b);
int quo (int a, int b);
int rem (int a, int b);
long int mpow (int a, int b);

int main(){
  int a, b;
  printf("Enter two integers: ");
  scanf("%d %d",&a, &b);
  printf("Quotient: %d\n",quo(a,b));
  printf("Remainder: %d\n",rem(a,b));
  printf("Sum: %d\n",sum(a,b));
  printf("Product: %d\n",prod(a,b));
  printf("Exponent: %ld\n",mpow(a,b));
  return 0;
}

int sum (int a, int b) { return a+b; }
int prod (int a, int b) { return a*b; }
int quo (int a, int b) { return a/b; }
int rem (int a, int b) { return a%b; }
long int mpow (int a, int b) { return pow(a,b); }
```