



B1- Unix and C Lab Seminar

B-CPE-100

Day 05

Recursivity

v2.0



Day 05

Recursivity

repository name: CPool_Day05_\$ACADEMICYEAR

repository rights: ramassage-tek

language: C

group size: 1



- Your repository must contain the totality of your source files, but no useless files (binary, temp files, obj files,...).
- Don't push your **main** function into your delivery directory, we will be adding our own. Your files will be compiled adding our **main.c** and our **my_putchar.c** files.
- You are only allowed to use the **my_putchar** function to complete the following tasks, but don't push it into your delivery directory, and don't copy it in *any* of your delivered files.
- If one of your files prevents you from compiling with *.c, the Autograder will not be able to correct your work and you will receive a 0.



Create your repository at the beginning of the day and submit your work on a regular basis!
The delivery directory is specified within the instructions for each task.
In order to keep your repository clean, pay attention to `gitignore`.



All of the day's functions must produce an answer in under 2 seconds.
Overflows must be handled (as errors).



Task 01

my_compute_factorial_it

Write an iterative function that returns the factorial of the number given as a parameter.
It must be prototyped the following way:

```
int my_compute_factorial_it(int nb);
```

In case of error, the function should return 0.

Delivery: CPool_Day05_\$ACADEMICYEAR/my_compute_factorial_it.c



$0! = 1$
if $n < 0$, $n! = 0$

Task 02

my_compute_factorial_rec

Write a recursive function that returns the factorial of the number given as a parameter.
It must be prototyped the following way:

```
int my_compute_factorial_rec(int nb)
```

In case of error, the function should return 0.

Delivery: CPool_Day05_\$ACADEMICYEAR/my_compute_factorial_rec.c



Task 03

my_compute_power_it

Write an iterative function that returns the first argument raised to the power p , where p is the second argument. It must be prototyped the following way:

```
int my_compute_power_it(int nb, int p);
```

Delivery: CPool_Day05_\$ACADEMICYEAR/my_compute_power_it.c



$n^0 = 1$
if $p < 0$, $n^p = 0$

Task 04

my_compute_power_rec

Write a recursive function that returns the first argument raised to the power p , where p is the second argument. It must be prototyped the following way:

```
int my_compute_power_rec(int nb, int p);
```

Delivery: CPool_Day05_\$ACADEMICYEAR/my_compute_power_rec.c

Task 05

my_compute_square_root

Write a function that returns the square root (if it is a whole number) of the number given as argument. If the square root is not a whole number, the function should return 0. It must be prototyped the following way:

```
int my_compute_square_root(int nb);
```

Delivery: CPool_Day05_\$ACADEMICYEAR/my_compute_square_root.c



Task 06

my_is_prime

Write a function that returns 1 if the number is prime and 0 if not.
It must be prototyped the following way:

```
int my_is_prime (int nb);
```

Delivery: CPool_Day05_\$ACADEMICYEAR/my_is_prime.c



As you know, 0 and 1 are not prime numbers.

Task 07

my_find_prime_sup

Write a function that returns the smallest prime number that is greater than, or equal to, the number given as a parameter.
It must be prototyped the following way:

```
int my_find_prime_sup (int nb);
```

Delivery: CPool_Day05_\$ACADEMICYEAR/my_find_prime_sup.c



Task 08

The n queens

Write a function that returns the number of possible ways to place n queens on a $n \times n$ chessboard without them being able to run into each other in a single move.
It must be prototyped the following way:

```
int count_valid_queens_placements(int n);
```

The output must be as follows:

```
Terminal
~/B-CPE-100> ./count_valid_queens_placements 1
1
~/B-CPE-100> ./count_valid_queens_placements 2
0
~/B-CPE-100> ./count_valid_queens_placements 3
0
~/B-CPE-100> ./count_valid_queens_placements 4
2
~/B-CPE-100> ./count_valid_queens_placements 5
10
```

Delivery: CPool_Day05_\$ACADEMICYEAR/count_valid_queens_placements.c



Damn it, this is recursion day!