C1 - Pool C

C-CPE-042

Day 08

Data structures and recursive programming

1.1

Day 08

repository name: pool_c_d08
repository rights: ramassage-tek

- The totality of your source files, except all useless files (binary, temp files, obj files,...), must be included in your delivery.

+ Instructions: Before You Go Further

- Turn in directory: **pool_c_d08**
- respecting the norm takes time, but is good for you. This way your code will respect the norm from the first written line to the last.
  Read carefully the norm documentation. You should type “alt+i” instead of “tab”
- You shall leave in your directory no other files than those explicitly specified by the exercises. If one of your files prevents the compilation with *.c, the robot will not be able to do the correction and you will have a 0.
- **Do not turn-in a main() function.**

+ Exercise 1: Structure (3pts)

Turn in: pool_c_d08/ex_01/struct.h
Function allowed for this exercise: -

Turn in a file 'struct.h' correctly protected against multiple inclusions. It must contain a structure named 's_my_struct'.
In this structure, you must have an integer named 'id' and a char pointer named 'str'.

+ Exercise 2: my_init (3pts)

Turn in: pool_c_d08/ex_02/struct.h
pool_c_d08/ex_02/ex_02.c
Function allowed for this exercise: -
Prototype: void my_init(t_my_struct *);

Now you must define a typedef so that the type of your structure changes from "struct s_my_struct" to "t_my_struct" to increase writing and readability.
Write a function 'my_init' in a file named 'ex_02.c' and using your structure from exercise 1. Your function will take a t_my_struct pointer in parameter. You should set the 'id' field of the structure to 0 and the 'str' field to NULL.

+ Exercise 3: my_abs (2pts)

Turn in: pool_c_d08/ex_03/abs.h
Function allowed for this exercise: -

Create a macro 'MY_ABS' in a file named 'abs.h' correctly protected against multiple inclusions. Your macro must take a number in parameter and return its absolute value.

+ Exercise 4: Combinaison (4pts)

Turn in: pool_c_d08/ex_04/abs.h
pool_c_d08/ex_04/struct.h
pool_c_d08/ex_04/ex_04.c
Function allowed for this exercise: strdup
Prototype: void my_init(t_my_struct *, int, const char *);

Take back your files from previous exercises and modify your function 'my_init'. It will now take three parameters: the same structure pointer as before, an integer and a char pointer.
The 'id' field will be initialized with the number given in parameter. You will apply your 'MY_ABS' macro to this number beforehand.
The 'str' field will be a copy of the char pointer in a newly allocated memory space in your function.

+ Exercise 5: my_power_it (2pts)

Turn in: pool_c_d08/ex_05/ex_05.c
Function allowed for this exercise: -
Prototype: int my_power_it(int, int);

Turn in a function 'my_power_it'. This function will take two integers in parameter, and must return an integer equal to the first parameter to the power of the second one (>= 0) using an iterative algorithm.

+ Exercise 6: my_power_rec (2pts)

Turn in: pool_c_d08/ex_06/ex_06.c
Function allowed for this exercise: -
Prototype: int my_power_rec(int, int);

Same as the previous exercise, but this time, your function must use a recursive algorithm.

+ EXERCISE 7: FIB_IT (2PTS)

Turn in: pool_c_d08/ex_07/ex_07.c
Function allowed for this exercise: -
Prototype: int fib_it(int);

Write an iterative function returning the nth rank of the fibonacci sequence.
The function will return the result or -1 in any error case.
https://en.wikipedia.org/wiki/Fibonacci_number

+ EXERCISE 8: FIB_REC (2PTS)

Turn in: pool_c_d08/ex_08/ex_08.c
Function allowed for this exercise: -
Prototype: int fib_rec(int);

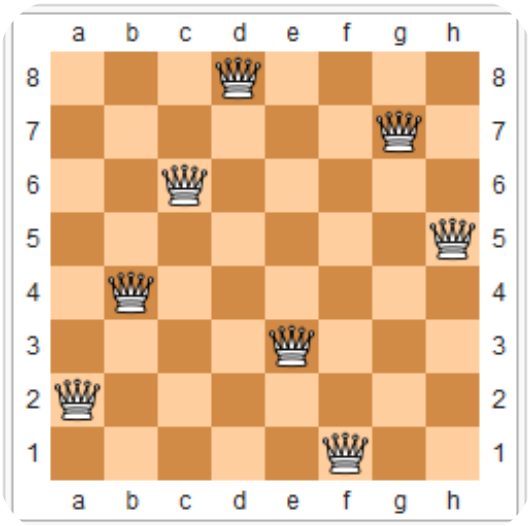
Write a recursive function returning the nth rank of the fibonacci sequence.
The function will return the result or -1 in any error case.
https://en.wikipedia.org/wiki/Fibonacci_number

+ Exercise Bonus: 8 Queens (5pts)

Turn in: pool_c_d08/ex_09/
Function allowed for this exercise: Everything on your dump.

Write a **software** "queens" that displays every possibility of placing 8 queens on a chessboard without them being able to turn each other's in a single move.

The display will be as below for the following solution example:



Example:

```
Terminal
~/C-CPE-042> cc *.c -o queens
~/C-CPE-042> ./queens
d8,g7,c6,h5,b4,e3,a2,f1
[...]
~/C-CPE-042>
```

In this example, we only have one solution to the "8 queens" problem. You need to find and display all possibilities to succeed the exercise.

There is a line break after each solution of the 8 queens' problem.

The order is important for the grinder, use its outputs to your advantage to find the right solution.