

Fibonacci Exercise

For this assignment you will write a program which calculates the n th Fibonacci number. If you need a refresher on Fibonacci numbers, see here:

https://en.wikipedia.org/wiki/Fibonacci_number

Organization

The only file that was included with this assignment is `README.txt`, which contains a link to these instructions. You will have to create the rest of the files yourself. See the last activity and the example code for guidance on how to organize your code and to write the Makefile.

Your program must be organized in the following files:

- `fibonacci.h` - Contains the prototype and documentation for your `fibonacci()` function, with an include guard.
- `fibonacci.c` - Contains the implementation of your `fibonacci()` function.
- `main.c` - Contains `main()` which gets the value of n from the user, uses your `fibonacci` function to calculate the n th Fibonacci number, and prints it out.
- `Makefile` - Builds the project to create the executable `fibonacci`. Creates the object file `fibonacci.o`, which is then compiled and linked with `main.c`. This file has no extension, it is simply called `Makefile`.

`fibonacci()`

Your `fibonacci()` function must have the following prototype:

```
unsigned long fibonacci(unsigned int n);
```

Above your prototype, write a comment which contains documentation for the function. As I have done for the functions you have written recently, describe what the function does, what the parameter is, and what the return value is.

Fibonacci numbers are non-negative and get large quickly, so `unsigned long` is a good choice for the return type (we are assuming a 64 bit `unsigned long`). n will be

non-negative as well but will not be nearly as large as the return value, so unsigned int will work fine for `n`.

The 0th Fibonacci number is 0, the 1st Fibonacci number is 1, and all the Fibonacci numbers after that are calculated by adding the previous 2 Fibonacci numbers together. So when `n` is 0 or 1 your function can immediately return the appropriate value. Otherwise you will use a loop which continually calculates the next Fibonacci number until you reach the `nth`. To accomplish this you need to keep track of the previous 2 values. In the body of the loop you will calculate the next value, and then update the values of the previous 2 values appropriately.

Include Guard

Every header file should have an include guard to prevent problems with multiple inclusion. So your `fibonacci.h` must begin with this:

```
#ifndef FIBONACCI_H
#define FIBONACCI_H
```

and end with this:

```
#endif
```

```
main()
```

In `main()` you will need to print out the value that `fibonacci()` returns. You need to use `%lu` as the format specifier to print out an `unsigned long` value.

Your `main()` will not be tested by git-keeper, but it will be graded for organization.

Submission

Push your submission to git-keeper. Make sure you do not add any extra files (such as the .o files and the executable) with git add.

In addition to passing the tests, you will also be graded on organization (your code must be organized as described above).

Your grade for this assignment will be out out of 20 points:

- 1 - Program Successfully Compiles
- 9 - Program meets organizational specifications above
- 10 - Program passes all tests