```
//    Author : Jose Goncalves
//    jose.braga.pt@gmail.com

//    Teacher : Nuno Peixoto
//    University : IPCA
//    17.03.2014


#ifndef GPIO_H_
#define GPIO_H_

#include <linux/i2c-dev.h>
#include <sys/ioctl.h>        // i2c
#include <sys/mman.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <stdio.h>

#include <sys/types.h>
#include <netinet/in.h>
#include <strings.h>
#include <fcntl.h>

#include <unistd.h>

class Gpio
{

// members (atributos)
private :
    int port;
    int status;
    int fd;
    volatile ulong *gpio;

// prototype of methods
public:

    Gpio();
    void Gpo(int,int) ;   // have to add 'inline' elese had error : multiple definition of 'Gpio(int,int)'
    int Gpi(int);
    ~Gpio();
} ;

#endif


// Since the definition of the constructor is not inline in the class definition,
// the compiler does not make it implicitly inline. Now, if you include the same file
// in multiple translation units (i.e. *.cpp files), the linker will produce exactly the error you see,
// because each of the *.cpp files will contain its own definition of the constructor without them being marked as inline functions.
// The solution is easy, just put a inline in front of the constructor declaration:

// ver http://www.tiexpert.net/programacao/c/funcoes-inline.php



// The former (using inline) allows you to put that function in a header file, where it can be included in multiple source files.
// Using inline makes the identifier in file scope, much like declaring it static.
// Without using inline, you would get a multiple symbol definition error from the linker.
// Of course, this is in addition to the hint to the compiler that the function should be compiled inline
// into where it is used (avoiding a function call overhead). The compiler is not required to act upon the inline hint.
```