Homework 6 Help

Setting Up:

You will need to use mercury 1 server for linker.c to work. To do that follow the commands below:

ssh username@mercury.cs.uml.edu

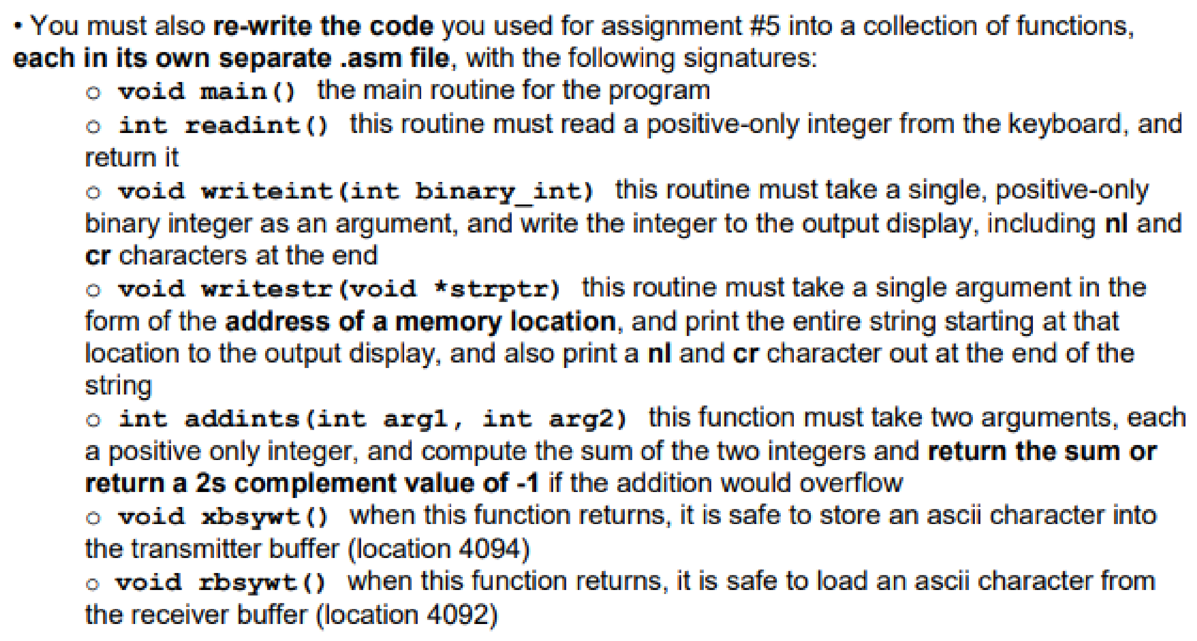
*put your password in

-Once logged in type: ssh mercury1

- Type YES and put your password in if it asks

This assignment is not very hard. All you need to do is combine some of the starter code and define a function in linker.c. All the code given is on the professor's website, but I have put it together since it might get confusing. I have attached linker.c to the discord message. All you need to do in linker.c is define int get_sym_val(char* symbol). The rest of the functions and code is already working. Next you need to create asm files for the functions below. You will also need mic1, prom_mrd.dat, masm_mrd. Use the commands below to download them. Make sure to include that period at the end

```
cp ~bill/cs305/a6_help_dir/masm_mrd .
cp ~bill/cs305/mic1 .
cp ~bill/cs305/a6_help_dir/prom_mrd.dat .
```



• You must also **re-write the code** you used for assignment #5 into a collection of functions, **each in its own separate .asm file**, with the following signatures:

- ○ **`void main()`** the main routine for the program
- ○ **`int readint()`** this routine must read a positive-only integer from the keyboard, and return it
- ○ **`void writeint(int binary_int)`** this routine must take a single, positive-only binary integer as an argument, and write the integer to the output display, including **nl** and **cr** characters at the end
- ○ **`void writestr(void *strptr)`** this routine must take a single argument in the form of the **address of a memory location**, and print the entire string starting at that location to the output display, and also print a **nl** and **cr** character out at the end of the string
- ○ **`int addints(int arg1, int arg2)`** this function must take two arguments, each a positive only integer, and compute the sum of the two integers and **return the sum or return a 2s complement value of -1** if the addition would overflow
- ○ **`void xbsywt()`** when this function returns, it is safe to store an ascii character into the transmitter buffer (location 4094)
- ○ **`void rbsywt()`** when this function returns, it is safe to load an ascii character from the receiver buffer (location 4092)

Create main.asm, readint.asm, writeint.asm, writestr.asm, addints.asm, xbsywt.asm, rbsywt.asm. Besides main.asm, the code for the rest of the files can be found inside hw5 starter code. You need to break down homework 5 stater code and put it inside the appropriate .asm file. main.asm will put together all the files. I have written the pseudo code for main.asm to help you get an idea.

Pseudo Code for main.asm:

```
Int main() {
    Turn on transmitter.
    Turn on receiver.
    Printf("Please enter 1-5 digit number followed by enter")
    Read number 1
     Printf("Please enter 1-5 digit number followed by enter")
     Read number 2
     Add number 1 and number 2 and put inside variable called sum (you can call it
                                                                        anything)
     Printf("The sum of the numbers is:")
     If sum == negative, go to overflow
     Print the sum
     Halt
 Overflow:
     Printf("Overflow occurred")
     halt
}
```

Running the Program:

1) Compile linker.c using the command below (only run this command if you make changes to linker.c otherwise do it once):
   ```
   gcc -o linker linker.c
   ```
2) Create .obj files from the .asm files using the commands below (only run these as you change the .asm file. You do not need to run them every time):
   ```
   ./masm_mrd -o <main.asm> main.obj
   ./masm_mrd -o <readint.asm> readint.obj
   ```

```
./masm_mrd -o <writeint.asm> writeint.obj
./masm_mrd -o <writestr.asm> writestr.obj
./masm_mrd -o <addints.asm> addints.obj
./masm_mrd -o <xbsywt.asm> xbsywt.obj
./masm_mrd -o <rbsywt.asm> rbsywt.obj
```

3) Create an executable file from the .obj files using the command below (run this every time you want to run your program):

   ```
   ./linker main.obj readint.obj writeint.obj writestr.obj addints.obj xbsywt.obj rbsywt.obj > program.exe
   ```

4) Run the executable file using the command below (run this every time you want to run your program):

   ```
   ./mic1 prom_mrd.dat program.exe 0 1024
   ```

Those commands will get program running. If it gets stuck try running again otherwise your .asm file code might be wrong. To get the consolidated file run the command below:

```
./linker -o main.obj readint.obj writeint.obj writestr.obj addints.obj xbsywt.obj rbsywt.obj > hw6.obj
```