CS 2110

Timed Lab 4

Due Date and Time

Day: Monday, October 24th
Time: Before the end of your assigned lab section

Policy

Submission

TURN IN THIS ASSIGNMENT ELECTRONICALLY USING T-SQUARE

SUBMISSIONS WHICH ARE LATE WILL NOT BE ACCEPTED.

EMAIL SUBMISSIONS WILL NOT BE ACCEPTED UNDER ANY CIRCUMSTANCES!

IF YOU FORGET TO HIT THE SUBMIT BUTTON YOU WILL GET A ZERO.

Questions

If you are unsure of what questions mean, the TA's will clarify them to the best of their ability. We will not be able to answer any questions about how to reach a solution to the timed lab questions. You should know how by now!

What's Allowed

- The assignment files
- Your previous homework and lab submissions
- The following 2 files from T-Square resources:
    - CS-2110 Resources / Assembly/REFERENCE SHEET.pdf
    - CS-2110 Resources / Assembly/PattPatelAppA.pdf
- Your mind
- Blank paper for scratch work

What's Not Allowed

- The Internet (except the T-Square Assignment page to submit)
- Any resource on T-Square that is not given in the assignment.
- Textbook or notes on paper or saved on your computer.
- Dropbox (If your harddrive crashes we will let you retake it).
- Email/IM
- Contact in any form with any other person besides TAs

  If you have any questions on what you may not use then assume you can't use it and ask a TA.

Other Restrictions

1. You may not leave the classroom until we have verified that you have submitted the lab. If you leave the classroom without submitting you will receive a zero.
2. **YOU MUST SUBMIT BY THE END OF YOUR LAB PERIOD**. Bear in mind that the clock on your computer may be a few minutes slow. You are supposed to have a full class period to work, and we are letting you use the 10 minutes between classes to make sure you have submitted your work. **WE WILL NOT ACCEPT LATE SUBMISSIONS**, be they 1 second or 1 hour late.
3. The timed lab has been configured to accept one submission. If you accidentally submit or submit the wrong version flag one of the TAs and we will reopen submission for you.

Violations

Failure to follow these rules will be in violation of the Georgia Tech Honor Code. **AND YOU WILL RECEIVE A ZERO** and you will be reported to Bill and the Office of Student Integrity.

We take cheating and using of unauthorized resources **VERY SERIOUSLY** and you will be in serious trouble if you are caught.

Remember

1. There is partial credit given, and some of it is just following the directions.
2. We allow you to use your homework assignment.
3. Please don't get stressed out during a timed lab. You have plenty of time; however, use your time effectively
4. Again, remember: Don't get stressed. Partial credit will be given for things you have done correctly. Do the best you can!
5. If you don't know something at least TRY. Do not just walk out of the lab or submit an empty file. Partial credit!
6. Remember what you can and can't use. If you don't know, then don't use it and ask a TA if you can use it. If we catch you with unauthorized resources we will give you a zero, so better to be safe than sorry.

Greatest Common Divisor

You will be writing a recursive implementation in LC3 Assembly to find the greatest common factor of two positive integers. You must carry-out your implementation according the LC3 calling convention.

Preconditions

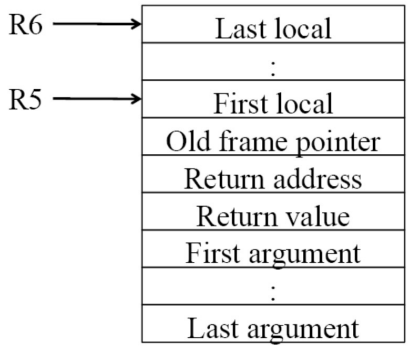
The A and B labels hold the numbers we want to find the greatest common factor of

Postconditions

The ANSWER label should now contain the greatest common factor of A and B

Calling Convention

A visual representation of the stack according to the LC3 calling convention is posted here for your reference:



Helpful Tips

The UDIV trap allows us to perform division and modulus. You call the UDIV trap with `TRAP x80`. Put A into R0 and B into R1. After executing `TRAP x80`, R0 will now hold A / B and R1 will now hold A % B.

Preconditions: R0 = A, R1 = B

Postconditions: R0 = A / B, R1 = A % B

For example: if R0 contained the value 4 and R1 contained the value 2, then, after executing UDIV, R0 would contain 2 and R1 would contain 0.

Pseudocode

```
int gcd(int a, int b) {

      if (a == 0) {
            return b;
      }
      if (b == 0) {
            return a;
      }

      int aModB = a % b;
      int bModA = b % a;

      if (a > b) {
            return gcd(b, aModB);
      }
      if (b > a) {
            return gcd(a, bModA);
      }
}
```

Testing

You are given a tester, which is a .xml file, that allows you to check many cases. If your code doesn't pass the test given in your assignment .asm file, you should debug your code through Complx. Then when you feel that your code works, try running it on the tester. Here is how:

1. Make sure gcd_test.xml and gcd.asm are in the same folder.
2. Navigate to that folder location on the terminal.
3. Enter the command: **lc3test gcd_test.xml gcd.asm**

The console will then output the results, what it is expected, and the actual output given. The .xml test *randomizes the contents of registers and all memory, and then loads your code* when it is inserted in the simulator.

Deliverables

1. gcd.asm

Note if your file is not named this you will lose points!

You may submit only the files listed above. We will not accept any internet links we want the files above and only these files!

Check over your submission after you submit it. If you submit the wrong file and leave the lab I will not be happy and we will grade what you submit so please check over what you submitted after you submit it!

Have fun and good luck!