CS224 - Spring 2024 - Lab #3 (Version 1, February 23, 15:43)

MIPS Assembly Language with Program as Data, Recursion, Linked Lists

Dates (TAs) - (Tutor(s))

Section 1: Mon, 4 Mar, 8:30-12:20 in EA-Z04 (Mustafa, Yiğit)
Section 2: Wed, 6 Mar, 13:30-17:20 in EA-Z04 (Mustafa, Yiğit)
Section 3: Tue, 5 Mar, , 13:30-17:20 in EA-Z04 (Kadri, Pouya) (Mehmet Can, 13:30 - 15:15)
Section 4: Fri, 15 Mar, Fri 8:30-12:20 in EA-Z04 (Onur, Soheil) (Bilal, Ece)
Section 5: Wed, 6 Mar, 8:30-12:20 in EA-Z04 (Kadri, Soheil)
Section 6: Fri, 15 Mar, 13:30-17:20 in EA-Z04 (Onur, Sepehr)

TA Full Name (email address: @bilkent.edu.tr)

M. Kadri Gofralılar (kadri.gofralilar)
Mustafa Yasir Altunhan (yasir.altunhan)
Onur Yıldırım (o.yildirim)
Pouya Ghahramanian (ghahramanian)
Sepehr Bakhshi (sepehr.bakhshi)
Soheil Abadifard (soheil.abadifard)
Yiğit Ekin (yigit.ekin)

Tutor Full Name (email address: @ug.bilkent.edu.tr)

Bilal Kar (bilal.kar)
Ece Beyhan (ece.beyhan)
Mehmet Can Bıyık (can.biyik)

Due to the Spring Break (March 7 Thursday - March 10 Sunday) Sections 1, 2, 3, and 5 will do the lab before the break in the week of March 4. On the other hand, Section 4 and 6 will do the lab after the break in the week of Mar 11.

The due date for the preliminary work is the same for all sections. See the course syllabus for the lab days.

Purpose: 1. This lab aims learning **1**. Practicing Von Neumann's stored program concept. **2.** Implementation of linked lists in assembly language. **3**. Implementation of recursion in MIPS.

If you use $t registers in subprograms (once or more) in Part 1 (Preliminary Work) you will lose 10 points. If you use $t registers in subprograms (once or more) in Part 2 (Lab Work) you will lose another 10 points.

Preliminary Work: 40 points

1. Instruction count (20 points)
2. Recursive division (20 points)

Lab Work: 60 points

1. Recursively print a linked list in reverse order (20 points)
2. Iteratively generate a copy of a linked list (20 points)
3. Recursively generate a copy of a linked list (20 points)

Important Notes for All Labs About Attendance, Performing and Presenting the Work

1. You are obliged to read this document word by word and are responsible for the mistakes you make by not following the rules.
2. Not attending to the lab means 0 out of 100 for that lab. If you attend the lab but do not submit the preliminary part you will lose only the points for the preliminary part.
3. Try to complete the lab part at home before coming to the lab. Make sure that you show your work to your TAs and answer their questions to show that you know what you are doing before uploading your lab work and follow the instructions of your TAs.
4. In all labs if you are not told you may assume that inputs are correct.
5. In all labs when needed you have to provide a simple user interface for inputs and outputs.
6. Presentation of your work
   You have to provide a neat presentation prepared in txt form. Your programs must be easy to understand and well structured.
   Provide following six lines at the top of your submission for preliminary and lab work (make sure that you include the course no. CS224, important for ABET documentation).

   CS224
   Lab No.
   Section No.
   Your Full Name
   Bilkent ID
   Date

   Please also make sure that your work is identifiable: In terms of which program corresponds to which part of the lab.
7. **If we suspect that there is cheating the lab grade will be zero and a disciplinary penalty is also possible. You cannot use ChatGPT code, it is classified as plagiarism. Note that MOSS is capable of detecting ChatGPT code.**

DUE DATE PRELIMINARY WORK: SAME FOR ALL SECTIONS

No late submission will be accepted. Please do not try to break this rule and any other rule we set.

a. Please upload your programs of preliminary work to Moodle by 8:30 on Monday March 4, 2024.

b. Please note that the submission closes sharp at 8:30 and no late submissions will be accepted. You can make resubmissions so do not wait for the last moment. Submit your work earlier and change your submitted work if necessary. Note that only the last submission will be graded.

c. Please familiarize yourself with the Moodle course interface, find the submission entry early, and avoid sending an email like "I cannot see the submission ınterface." (As of now it is not yet opened.)

d. Do not send your work by email attachment they will not be processed. They have to be in the Moodle system to be processed.

e. Use filename **StudentID_FirstName_LastName_SecNo_PRELIM_LabNo.txt** Only a NOTEPAD FILE (txt file) is accepted. Any other form of submission receives 0 (zero).

DUE DATE PART LAB WORK: (different for each section) YOUR LAB DAY

a. You have to demonstrate your lab work to your TA for grading. Do this by **12:00** in the morning lab and by **17:00** in the afternoon lab. Your TAs may give further instructions on this. If you wait idly and show your work last minute, your work may not be graded.

b. At the conclusion of the demo for getting your grade, you will **upload your Lab Work** to the Moodle Assignment, for similarity testing by MOSS. See below for the details of lab work submission.

c. Try to finish all of your lab work before coming to the lab, but make sure that you upload your work after making sure that it is analyzed by your TA and/or you are given the permission by your TA to upload.

Part 1. Preliminary Work (40 points)

1. **instructionCount (20 Points):** Write a non recursive subprogram that counts the total number of add (e.g., "add $t0, $t1, $t2"), ori (or immediate), lw (load word) instructions and returns this sum as its result. The address of the first instruction to be considered in counting is given in $a0, and the address of the last instruction to be considered in counting is given in $a1. Your main program should invoke this subprogram twice: first for the main (itself), and for the second time for the subprogram.

As you see, in this program your main program becomes an input data and in the second call the subprogram becomes its own data (like an medical operator operating itself). Provide meaningful output in main so that you and your TA can validate the correctness of your program.

Note that, in our practice to simplify things, we write MIPS programs in a single source file; therefore, all labels are at the global scale. For example, if you label the first instruction of your subprogram with the label L1, and its last instruction with the label L2, you can pass these addresses to the subprogram with load address instructions in main (la $a0, L1; la $a1, L2).

In MARS settings find "Self-modifying code" and mark that option so that you can access the machine instructions as if they are data.

In the lab folder see the program that displays its own instructions in hexadecimal.

2. **RecursiveDivision (20 points)**: Write a **recursive** MIPS subprograms that finds the division of a positive number by another positive number by successive subtractions. Your main program must

provide a user interface: It should get a pair of numbers from the user provide the answer, and should ask the user if the user wants to continue, if so it should ask the next pair of numbers, etc.

Part 2. Lab Work (60 points)

In the following parts first construct a linked list that contains ten numbers (1, 2, 3, ... 10 in this order). For the linked list construction use the program provided to you in the Moodle lab folder that contains this lab assignment (modify it as needed). Display the linked list again with the program you are given to make sure that the linked list is constructed properly. After that do what you are asked in each part below. You must provide simple easy to follow program output with proper messages for the user (your TA). Implement each part in a separate source program for easy grading and to prevent confusion during grading.

1. **DisplayReverseOrderRecursively (20 points)**: Write a **recursive** program to display the elements of a linked list in reverse order (10, 9, ... 1). The list head is passed in $a0. Provide a simple user interface.

2. **DuplicateListIterative (20 points)**: Write a **non recursive** subprogram to duplicate a linked list. When called $a0 points to the original list. It returns the new list head in $v0. Provide a simple user interface.

3. DuplicateListRecursive (20 points): Write a **recursive** subprogram to duplicate a linked list. When called $a0 points to the original list. It returns the new list head in $v0. Provide a simple user interface.

Part 3. Submit Lab Work for MOSS Similarity Testing

1. Submit your Lab Work MIPS codes for similarity testing to Moodle.
2. You will upload one file. Use filename **StudentID_FirstName_LastName_SecNo_LAB_LabNo.txt**
3. Only a NOTEPAD FILE (txt file) is accepted. No txt file upload means you get 0 from the lab. Please note that we have several students and efficiency is important.
4. *Even if you didn't finish, or didn't get the MIPS codes working, you must submit your code to the Moodle Assignment for similarity checking.*
5. Your codes will be compared against all the other codes in the class, by the MOSS program, to determine how similar it is (as an indication of plagiarism). So be sure that the code you submit is code that you actually wrote yourself ! You are not allowed to use web resources that solves the assigned programs.
6. The effectiveness of MOSS for chatbot code is quite good, with a detection rate of much higher than 50%. (The answer is provided by chatbot.)

Part 4. Cleanup

1. After saving any files that you might want to have in the future to your own storage device, erase all the files you created from the computer in the lab.
2. When applicable put back all the hardware, boards, wires, tools, etc where they came from.
3. Clean up your lab desk, to leave it completely clean and ready for the next group who will come.

LAB POLICIES

1. You can do the lab only in your section. Missing your section time and doing in another day is not allowed.
2. The questions asked by the TA will have an effect on your lab score.
3. Lab score will be reduced to 0 if the code is not submitted for similarity testing, or if it is plagiarized. MOSS-testing will be done, to determine similarity rates. Trivial changes to code will not hide plagiarism from MOSS—the algorithm is quite sophisticated and powerful works for ChatGPT code too. Please also note that obviously you should not use any program available on the web, or in a book, etc. since MOSS will find it. The use of the ideas we discussed in the classroom is not a problem.
4. You must be in lab, working on the lab, from the time lab starts until your work is finished and you leave.
5. No cell phone usage during lab. Internet usage is permitted only to lab-related technical sites.