

## CS224 - Spring 2023 - Recitation #3 (Version 1: March 20, 8:27)

### Program as Data, Recursion, and Linked Lists in MIPS Assembly Language

#### Dates (TAs) - (Tutor(s))

Section 1: Mon, 27 Mar, 8:30-12:20 in EA-Z04 (Deniz, Ece) - (Berkan)  
Section 2: Wed, 29 Mar, 13:30-17:20 in EA-Z04 (Sepehr, Pouya) - (Berkan, Ege )  
Section 3: Tue, 28 Mar, , 13:30-17:20 in EA-Z04 (Onur, Utku) - (Talay )  
Section 4: Fri, 31 Mar, Fri 8:30-12:20 in EA-Z04 (Sanaz, Soheil) - (Eren )  
Section 5: Wed, 29 Mar Feb, 8:30-12:20 in EA-Z04 (Navid, Onur) - (Yarkin )  
Section 6: Fri, 31 Mar, 13:30-17:20 in EA-Z04 (Sanaz, Soheil) - (Yarkin )

#### TA Name (email address: @bilkent.edu.tr): Section (hours)

Ece Kunduracıoğlu (e.kunduracioglu@): Section 1 (Mon 08:30-10:30)  
Deniz Uzel (deniz.uzel@): : Section 1 (Mon 08:30-10:30)  
Navid Ghamari (navid.ghamari@): Section 2 (Wed 13:30-15:30)  
Onur Yıldırım (o.yildirim@): Section 3 (Tue 13:30-17:30)  
Pouya Ghahramanian ( ghahramanian@): Section 2 (Wed 13:30-17:30)  
Sanaz Gheibuni (sanaz.ghuibuni@): Section 4 (Fri 08:30-12:30), Section 6 (Fri 13:30-15:30)  
Sepehr Bakhshi (sepehr.bakhshi@): Section 2 (Wed 13:30-17:30)  
Soheil Abadifard (soheil.abadifard@): Section 4 (Fri 08:30-12:30), Section 6 (Fri 13:30-15:30)  
Utku Gülgeç (utku.gulgec@): Section 3 (Tue 13:30-17:30)

#### Tutor Name ( email address: @ug.bilkent.edu.tr): Section (hours)

Berkan Şahin (berkan.sahin@): Section 1 (Mon 08:30-10:20) & Section 2 (Wed 13:30-15:20)  
Hasan Ege Tunç (hasan.tunc@): Section 2 (Wed 13:30-17:20)  
Atak Talay Yücel (talay.yucel@): Section 3 (Tue 13:30-17:20)  
Mehmet Eren Balasar (eren.balasar@): Section 4 (Fri 08:30-12:20)  
Hasan Yarkin Kurt (yarkin.kurt@): Section 5 (Wed 08:30-10:20) & Section 6 (Fri 13:30-15:20)

**Purpose:** 1. Understanding the Von Neumann stored program concept. 2. Fundamentals of dynamic data structure construction in assembly language. 3. Implementation of recursion in assembly language.

**Important Implementation Notes and Requirements for Recitation 3:** You are not allowed to use \$t registers in the subprograms: Use \$s registers to get used to the MIPS software development traditions. When you enter to a subprogram save \$s registers you use in the subprogram to the stack. When necessary save \$ra register too. Remember and use the rules of passing parameters (use \$a0, \$a1, \$a2, \$a3 registers as needed) to subprograms and returning results (using \$v0, \$v1) to the caller.

You are free to use the linked list construction and linked list printing subprograms provided in the related recitation folder. You may modify them according to your needs. If you need you may define and use additional subprograms other than the ones defined in this document. Make sure that you work is presentable and easy to follow.

For all parts provide a simple console interface that displays messages to the user to get inputs.

In all programs you may assume that user inputs are correct.

You are required to use proper syntax and meaningful variable and label names. Make sure that you explain your code and use of registers.

## Summary

### Preliminary Work (50 points)

Understanding stored program concept (program as data). Learning the principles of recursive programming and the implementation of dynamic link list data structure.

### Recitation Work (50 points)

More experience with linked lists and recursive programming.

**Online Recitation Class: (Note try and study recitation part at home before coming to the recitation. Make sure that you show the recitation work to your TA before uploading during online recitation class.)**

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

1. This document defines an online class activity.
2. You are obliged to read this document word by word and are responsible for the mistakes you make by not following the rules.
3. Not attending to the recitation means 0 out of 100 for that recitation. If you attend the recitation but do not submit the preliminary part you will lose only the points for the preliminary part.
4. Try to complete the recitation part at home before coming to the recitation. 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 recitation work and follow the instructions of your TAs.
5. In all recitations if you are not told otherwise, you may assume that inputs are always correct.
6. In all recitations when needed you have to provide a simple user interface for inputs and outputs.
7. 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 recitation work (make sure that you include the course no. CS224, important for ABET documentation).

CS224

Recitation 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 recitation.

- 8. If we suspect that there is cheating we will send the work with the names of the students to the university disciplinary committee. 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 Mar 27, 2023.
- 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 interface." (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\_RecitationNo.txt** Only a NOTEPAD FILE (txt file) is accepted. Any other form of submission receives 0 (zero).

## **DUE DATE PART RECITATION WORK: (different for each section) YOUR RECITATION DAY**

- a. You have to demonstrate your recitation work to your TA for grading. Do this by **12:00** in the morning recitation and by **17:00** in the afternoon recitation. 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 Recitation Work** to the Moodle Assignment, for similarity testing by MOSS. See below for the details of recitation work submission.
- c. Try to finish all of your recitation work before coming to the recitation, 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 (50 points)**

1. **RegisterCount (20 points):** Write a subprogram that counts the number of times a given register number (1 to 31) is used in all instructions of the subprogram. For example, in the instruction `add $t0, $t0, $t1` the register number 8 (`$t0`) is used twice. In the instruction `lw $t1, 0($t0)` the register number `$t0` is used once. The input is the register number and the output is the number of times that register is used in the subprogram.

For this program to run in MARS Settings SelfModifyingCode option must be turned on. The main program must provide the user interface and it should stop when the user enters a number not within the range of 1 to 31. To understand how to access instructions as data see the program printYourself available in the recitation folder.

2. **recursiveDivision (15 points):** Write a **recursive** MIPS subprogram that performs integer division of two positive numbers by successive subtractions. As a challenge you may want to implement it to return the quotient in \$a0 and remainder in \$a1.
3. **CopyAllExcept\_x (15 points):** Write a subprogram to copy all entries of a linked list to a newly created linked list in the same order except a value which is given as a parameter. Linked list contains integer numbers. Subprogram returns the list head of the newly created linked list (in \$v0) and its size in terms of number of nodes (in \$v1). For example if the original linked list <1, 3, 8, 7, 7, 10, 3> (data values are shown) and the value not to be included is 3, the newly created linked list becomes <1, 8, 7, 7, 10> and its size is in \$v1 is 5. Provide the necessary user interface.

## Part 2. Recitation Work (50 points)

1. **DeleteMultipleEntries (20 points):** Write a subprogram to delete multiple values from a sorted linked list. For example, after executing this subprogram <1, 2, 2, 5, 5, 5, 7> becomes <1, 2, 5, 7>
2. **PrintInReverseOrder (15 points):** Write a **recursive** subprogram to print the contents of a linked list in reverse order. For example, <8, 7, 3, 15> is printed as 15, 3, 7, 8.
3. **ReverseLinkedList (15 points):** Write a **recursive** subprogram to reverse a linked list. For example, if the linked list is originally <8, 7, 3, 15> it becomes <15, 3, 7, 8>. Note that you will use the same nodes and reverse the direction of the pointers.

## Part 3. Submit Recitation Work for MOSS Similarity Testing

1. Submit your Recitation Work MIPS codes for similarity testing to Moodle.
2. You will upload one file. Use filename  
**StudentID\_FirstName\_LastName\_SecNo\_RECITATION\_RecitationNo.txt**
3. Only a NOTEPAD FILE (txt file) is accepted. No txt file upload means you get 0 from the recitation. 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: This applies if you do your work in a publicly accessible computer**

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 recitation.
  2. When applicable put back all the hardware, boards, wires, tools, etc where they came from.
  3. Clean up your recitation desk, to leave it completely clean and ready for the next group who will come.
- 

#### **RECITATION POLICIES**

1. You can do the recitation 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 recitation score.
3. Recitation 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 recitation, working on the recitation, from the time recitation starts until your work is finished and you leave.
5. No cell phone usage during recitation.
6. Internet usage is permitted only to recitation-related technical sites.
7. **For the Earthquake related cases we follow the policies set by the university administration. Note that they require official documentation. If applicable to you please follow the rules set by the university.**