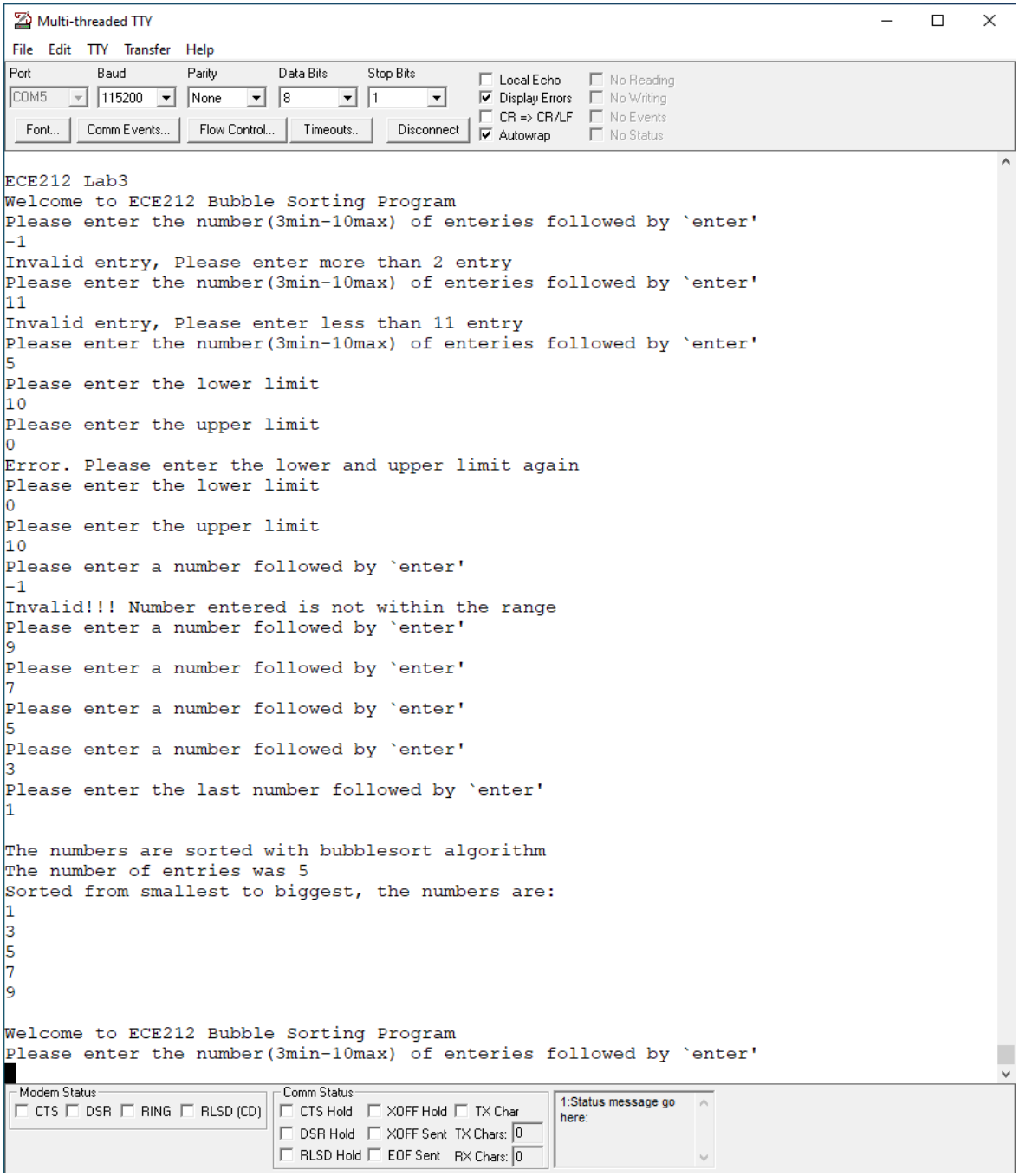
Lab 3: Introduction to Subroutines



Lab Dates

Refer to the schedule on the ECE212 Laboratories page for the latest schedule

Introduction

In this lab, the students will be learning how to divide up task by writing subroutines for a bubble sort algorithm program. Good subroutines are expected to be completely contained pieces of code with one entry point and one exit point. In addition, either the caller (the main program) or the callee (the subroutine) must preserve register values to ensure proper execution of code. In ECE212, the callee method will always be used. Your subroutines cannot modify any registers except for registers designated for return values. You must save and restore any registers used in your subroutines.

Objectives:

1. To gain experience using the STACK(Push and Pop).
2. To gain experience in dividing up existing code into subroutines.
3. To gain experience in calling subroutines/functions.
4. To learn the basic parameter passing techniques.

Prelab and Preparation:

- Read the lab prior to attending your lab session. Please only attend the section that you are registered in.
- Do the online prelab Quiz **individually** and submit it before the deadline
- Prepare rough drafts of the flow chart(s)/pseudocode for each subroutine(3 in total). You do not neet to use a CAD tool for this. The final CAD version will be submitted in your lab report.

Lab Work and Specifications

1. Download the template files
    - main_L432KC.c
    - Lab3_L432KC.s
    - Lab3a_L432KC.s
    - Lab3b_L432KC.s
    - Lab3c_L432KC.s

Specifications

If you recall from the Lab1, the 'main.c' is the standard project template that is used to initialize and call standard functions/subroutines including our 'AssemblyProgram'. Do not modify any of the parameters in this file. 'Lab3_L432KC.s' has been provided to call your subroutines. Do not modify any of the parameters in this file. Lab3a_L432KC.s, Lab3b_L432KC.s and Lab3c_L432KC.s are provided for you to write your subroutines. Each Subroutine has certain passed parameters and certain conditions. Carefull attention should be paid. A more detail description is provided below.

The requirements on each subroutine are indicated below.

1. WelcomePrompt

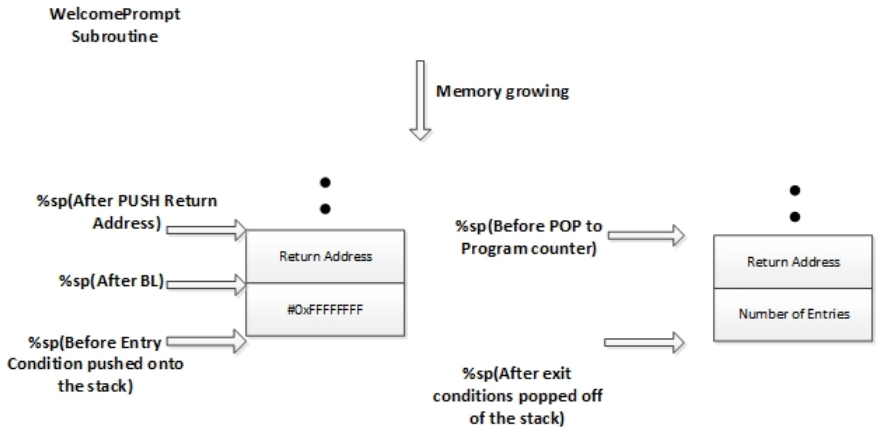
   In the Lab3a_L432KC.s template file, you are asked to write a subroutine called *WelcomePrompt* that prompts the user to enter the number of entries for a bubble sort program followed by asking for the lower and upper limit boundaries. This will set the range for all valid numbers allowed. Finally, the user will enter the numbers that will be sorted from the keyboard.

   The number of entries(word-32bits) will be passed out through the stack(Look for the flag on the stack and replace it). The numbers entered that will be sorted are stored at some memory location given in register R0(ie, 0x20001000). Note, we can change the value in R0 so please don't hard code your answer to it. You must use whatever value is given in R0. Each number will occupy one word(32bits) of memory.

   Certain restrictions have been placed on the number of entries and the range of the numbers. The number of entries entered by the user must be between **3 to 10**(min 3, max 10). The values entered for the bubble sort algorithm must be between the user specified range. If the lower and upper range entered are incorrect(for example, lower limit = 9, upper limit = 0), re-prompt with an error message and allow the user to input again. Please note that the lower limit cannot be the same value as the upper limit. Finally, for the numbers used in the sorting program, if the number entered is not within the range, re-prompt with an error message and allow the user to input again. Your program should also flag when the last number is to be entered.

   Stack Entry Condition = memory space allocated for the number of entries on stack(word-32bits)

   Stack Exit Condtion = number of entries on stack(word-32bits)



Example - Your program should look something like this:

Welcome String

- Display a welcome string on startup.
  Welcome to ECE 212 Bubble Sorting Program

User Input Prompts

- Display a string prompting the user to enter the number of entries.
  Please enter the number(3min-10max) of entries
- Prompt the user to enter the lower and upper limit separately.
  Please enter the lower limit
  Please enter the upper limit
- Display a string prompting the user to enter a number.
  Please enter a number
- Display a string prompting the user to enter the last number.
  Please enter the last number

Error Messages

- Display a string indicating invalid entry if an improper value was entered.
  Invalid entry, Please enter more than 2 entry
  Invalid entry, Please enter less than 11 entry

- Display a string indicating the lower and upper limit is incorrect.
  Error. Please enter the lower and upper limit again

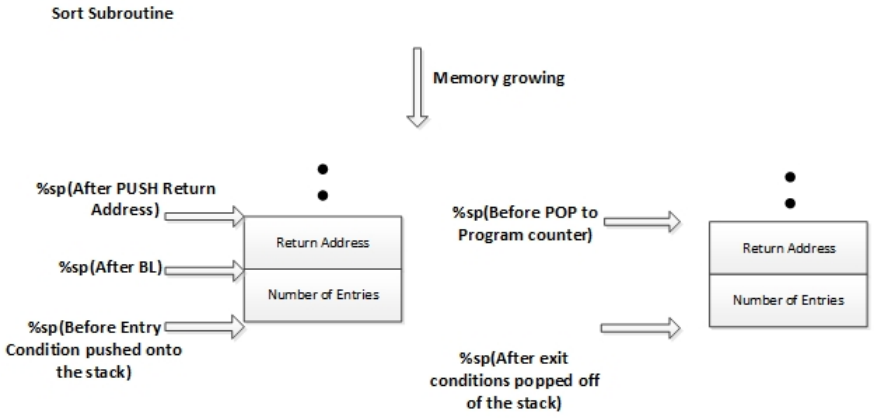
- Display a string indicating the number is not within the lower and upper limit.
  Invalid!!! Number entered is not within the range

2. Sort

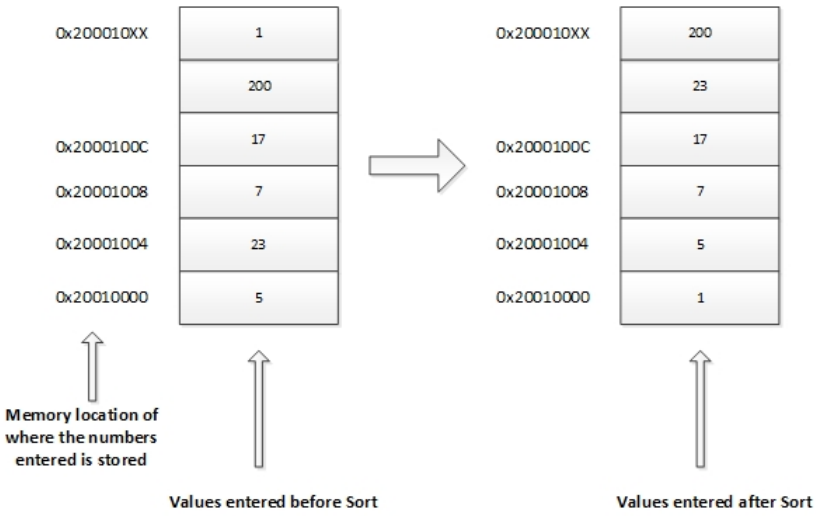
   In the Lab3b.s template file, you are to code a subroutine called *Sort* that implements the bubble sort alogorith. The value of the number of entries(word-32bits) is passed in through the stack. The memory location where the entries are stored is once again provided in the register R0.

   Stack Entry Condition = Number of entries(word-32bits) on the stack.

   Stack Exit Condtion = Number of entries(word-32bits) remain on the stack



   The figure below illustrates an example of the sort with 6 entries stored starting at memory location given in R0(ie, 0x20001000).

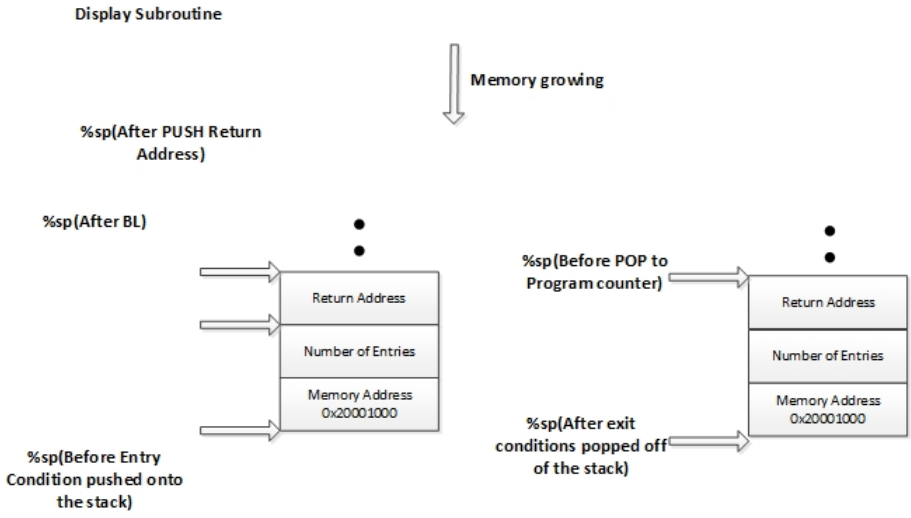


3. Display

   In the Lab3c.s template file, you are to code a subroutine called *Display* that displays the numbers from lowest to highest. The value of the number of entries along with the memory location where the numbers are stored are passed through the stack.

   Stack Entry Condition = number of entries(word-32bits) and address of where the entries are stored(word-32bits) on the stack

   Stack Exit Condtion = number of entries(word-32bits) and address of where the entries are stored(word-32bits)remain on the stack



Example - Your Display should look something like this:

- The number of entries should be indicated
  The number of entries entered was XX
- Display a string indicating the numbers are sorted from smallest to largest.
  Sorted from smallest to biggest, the numbers are:
- Display the numbers.
- Display a string ending the program.
  Program ended

Note: For each subroutine, only the information provided can be used. No assumptions can be made.

Provided Subroutines

1. printf

   Prints a string to the monitor. No carriage return or linefeed is invoked after calling this function.

   Entry Condition = address of the string in register R0

   Exit Condition = None

Example of pseduocode usage:
move StringLabel address into register R0
Jump to printf subroutine

2. cr

   A function that generates a carriage return and linefeed

   Entry Condition = None

   Exit Condition = None

   Example of pseduocode usage:
   Jump to cr subroutine

3. value

   A function that prints a decimal number to the monitor with no carriage return and linefeed.

   Entry Condition = decimal value in register R0

   Exit Condition = None

   Example of pseduocode usage:
   Move Decimal value into register R0
   Jump to value subroutine

4. getstring

   A function that prompts the user to enter a number from the keyboard. Valid entries are all numbers (negative and positive)

   Entry Condition = None

   Exit Condition = Decimal number stored in Register R0

   Example of pseduocode usage:
   Jump to getstring subroutine

   Note: For simplicity, a check routine for a proper number entry has been provided. Invalid characters are rejected and cleared from the buffer and the user is prompted to re-enter a proper number.

Data Section

Strings are stored in the .data section in your template files. For example:

Welcome:
.string "This is the welcome string"
This will store the string label Welcome at a certain memory location. At that memory location, the string will be stored in hexadecimal. Each letter will occupy one memory block(1byte).

Questions

1. Is it always necessary to implement either callee or caller preservation of registers when calling a subroutine. Why?
2. Is it always necessary to clean up the stack. Why?
3. If a proper check for the getstring function was not provided and you have access to the buffer, how would you check to see if a valid # was entered? A detailed description is sufficient. You do not need to implement this in your code.

Marking Scheme

Lab 3 is worth 25 % of the final lab mark. Please view the Marking Sheet for this lab to ensure that you have completed all of the requirements of the lab. The Marking Sheet also provides a limited test suite in the demo section for you to think about. Make use of it! When writing the report, please follow the Report Writing Guidelines document uploaded to Eclass.

Demo and Report

The **report and demo due dates** are given on the ECE212 Laboratories page on Eclass. Note that you have one week from the dating of your prelab to complete the demo.

The report must be submitted using the submission link on Eclass before **4:30 P.M. Do not be late** on your scheduled due date.

Late Submissions

There are late submission penalties for the Demo(-10%) and Report(-10%) per day after the scheduled due date.