Indian Institute of Technology (IIT-Kharagpur)

AUTUMN Semester, 2021

COMPUTER SCIENCE AND ENGINEERING

Computer Organization and Architecture Laboratory

MIPS Assignment 2

September 8, 2021

AIM: To get proficient in handling arrays in MIPS, writing function subroutine in MIPS, allocating variables dynamically on the stack, passing parameters to functions by value, passing (local) arrays to functions by their addresses.

INSTRUCTIONS: Make one submission per group in the form of a single zipped folder containing your source code(s). Name your submitted zipped folder as Assgn_3_Grp_GroupNo.zip and (e.g. Assgn_3_Grp_25.zip). Inside each submitted source files, there should be a clear header describing the assignment no., problem no., semester, group no., and names of group members. Liberally comment your code to improve its comprehensibility.

Question 1

Write a complete MIPS-32 program that -

1. Reads two 16-bit signed integers (encoded in 2's complement form) with the prompt – "Enter first number:" and "Enter second number:".
2. After the two input numbers are collected from the user, there should be sanity checking to ensure that they are within the proper numerical range.
3. Determine the product of these two numbers using the **Booth's Multiplication Algorithm** as depicted in Figure 1.
4. At the end of your program, print the calculated product with the message – "Product of the two numbers are: ".

Your program should have a procedure call multiply_booth, with the two input arguments available in registers $\$a0$ and $\$a1$, and the final product available in register $\$v0$.

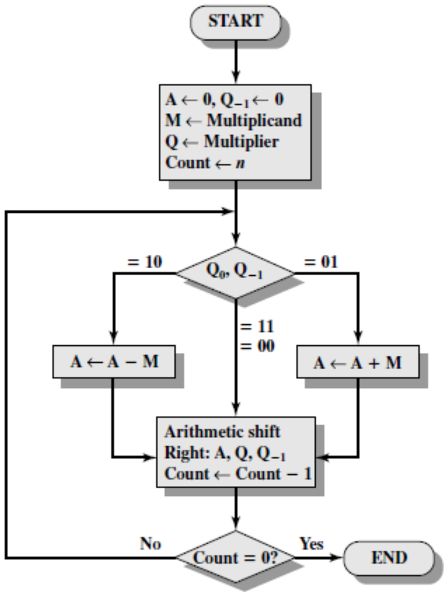

Figure 1: Booth's Algorithm

Question 2

Write a complete MIPS-32 program that -

1. Reads an array of ten integers from the user (can also be negative). These numbers are collected from the input console using a loop and stored in the memory in an array called 'array'. **Do not store the numbers as scalars in ten different non-contiguous locations or in ten different registers.**

2. Reads an integer 'k' with the prompt: "Enter the value of k: ". Write a function named **find_k_largest** that finds the k^{th} largest number in the above array. Your program should also check whether $k > n$. In such case, your program should display proper error message.

3. Write a function named **sort_array** in order to sort the input array before finding the k^{th} largest number. You have to implement Algorithm 1 as given below to sort the array. Pass the address of the 1-D array and the required parameters while implementing the function. After sorting, print the sorted array on the console with a proper message.

4. Thereafter, print the k^{th} largest number from the sorted array with suitable messages.

Algorithm 1 sort_array (Sort the array denoted as A with n numbers)

```
1: for $j = 2$ to $n$
2:     $temp = A[j]$;
3:     // Insert $A[j]$ to the sorted
         sequence $A[1..j-1]$
4:     $i = j - 1$;
5:     while $i > 0$ and $A[i] > temp$
6:         $A[i+1] = A[i]$;
7:         $i = i - 1$;
8:     $A[i+1] = temp$
```

Question 3

Write a complete MIPS-32 program that -

1. Prompts the user for four positive integers m, n, a r as "Enter four positive integers m, n, a and r: ".
2. Allocates space for an $m \times n$ integer array A and an $n \times m$ integer array B on the stack.
3. Populates the array A in a row major fashion using a Geometric Progression (GP) series with initial value a and common ratio r.
4. Print the elements of matrix A.
5. Computes the transpose matrix of the matrix A in the $n \times m$ matrix B.
6. Prints the elements of B finally.

Follow these implementation-level constraints while writing your code. Write the following functions:

1. "initStack" : Initialise the stack pointer (\$$sp$) and frame pointer (\$$fp$).
2. "pushToStack" : This function takes one argument as input (in \$$a0$) and push it to the stack.
3. "mallocInStack" : This function allocates space for $m \times n$ (stored in \$$a0$) memory locations (each of 4 bytes for each integer) in the stack and returns the first address (\$$v0$).
4. "printMatrix" : This function takes three parameters- the positive integers m (in \$$a0$), n (in \$$a1$) and the address of a two-dimensional $m \times n$ integer array A (in \$$a2$) storing the matrix in row major form. It prints the elements of A in a row major manner.
5. "transposeMatrix" : The function takes (positive) integers m and n and addresses of two integer arrays A and B. It is to compute the transpose matrix of the matrix A and store in the $n \times m$ matrix B.

If required, you can write additional functions as well, but with proper comments and descriptions.