1 Project 3: MIPS Assembly Code Programming Using MARS Tool – Additional Coding Assignment (CS 7381 only)

10 points

Students enrolled in CS 7381 (graduate version of the course) have an additional MARS programming assignment.

For this exercise, you will write another MIPS assembly code program. This time, you will create and run MIPS code for the following high-level language (HLL) code:

C++ Version

```
int j, k, n, x;

x = 1;
n = 4;
j = n;

// outer loop with j

do
{
    k = n;

    // inner loop with k

    do
    {
        x = x + 2 * j + 1;
        cout << x << " ";
        k--;
    } while (k > 0);

    j--;

} while (j > 0);
```

Python Version

```
x = 1
n = 4
j = n

# outer loop with j

while True:
    k = n

    # inner loop with k

    while True:
        x = x + 2 * j + 1
        print(x, end=" ")
        k -= 1
        if (k == 0):
            break
    j -= 1
    if (j == 0):
        break
```

1. Save the program as an *.asm file – use the first initial of your first name and the first 4 letters of your last name, then the number 2 (to distinguish from your code for Project 2). For example, my file submission name would be **tmani2.asm**.
2. **Take a screen shot of the MARS console (including Run/IO section) so that the grader can view your results.** If you take a screen shot of the entire console, please crop and enlarge the Run/IO section so that it is readable.
3. **Submit the actual *.asm file so that the grader can run it.**
4. Turn in your code and output (screen shot) for credit. **Make sure that your name appears on <u>both</u> documents that you submit so that you can get proper credit for your work.**

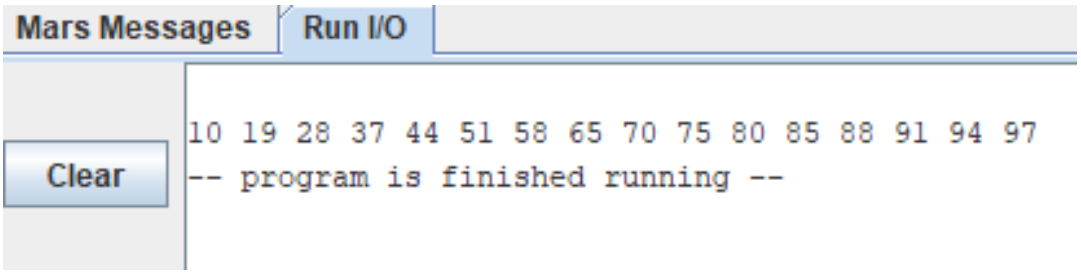
2 SOLUTION

2.1 Code (.asm)

An example of the MIPS assembly code solution is the following:

```
################################
# tmani2.asm
# Theodore Manikas
# 2022 Nov 30
# CS/ECE 5/7381 Spring 2023
################################

.data
x:          .word 1                 # variable x; init to 0
n:          .word 4                 # constant value for variable n
two:        .word 2                 # constant 2
space:      .asciiz  " "            # space to insert between numbers
#

.text
            lw    $t0, x            # variable x
            lw    $t1, n            # variable n
            lw    $t2, n            # variable j
            lw    $t3, n            # variable k
            lw    $t4, two          # constant 2


LoopJ:      add   $t3, $t1, $zero        # assign k <= n

LoopK:      mul   $t5, $t4, $t2          # 2 * j
            addi  $t5, $t5, 1          # 2 * j + 1
            add   $t0, $t0, $t5          # x = x + 2 * j + 1

            li    $v0,1                  # print x (integer value)
            add   $a0, $t0, $zero
            syscall

            la    $a0, space     # load address of spacer
            li    $v0, 4         # specify Print String service
            syscall                # print the spacer string

            subi  $t3,$t3,1        # k--
            bgtz  $t3, LoopK       # repeat loop if k>0

            subi  $t2,$t2,1        # j--
            bgtz  $t2, LoopJ       # repeat loop if j>0



            li   $v0, 10        # system call for exit
                 syscall         # we are out of here.
```

2.2 Results


Mars Messages
Run I/O
Clear
10 19 28 37 44 51 58 65 70 75 80 85 88 91 94 97
-- program is finished running --