A2 (25 marks)

Objectives: The MARS Environment; Basics of Assembly Programming Language

Q1. **The MARS environment**: (10 marks)

a) Locate and start MARS on your windows machine.
b) Download this program file Q1.asm and save it to your machine (e.g., in your F: drive).
c) Load `Q1.asm` into MARS. Assemble the file (F3) and then run it (F5). If everything goes well, you will see in the Console window (bottom) the following output: "Hello, Your ID =" . Type in any text string (up to 12 characters), press the "Enter" key, and observe the output.
d) Play with this program several times to get an idea what it does.
   - **Q 1.1: What does this program do?** (1 mark)
e) Now, reset MIPS registers and memory (F12). Press the button Step (F7) to trace the program. Observe the change of the registers $PC, $t0-$t2, $s0, $v0, $a0, and $a1. These registers are located in the right pane (Registers tab) in the MARS window. Answer the following questions (the content of the Data Segment is displayed in the middle pane of the MARS window):
   - **Q 1.2: What are the memory addresses of the data labeled `len`, `buffer`, and `str`?** (6 marks)
   - **Q 1.3: Will you answer to Question 2 remain the same if the directive ".align 4" is commented out? Why?** (3 marks)

Submit your answer to the above three questions in a file named A2_Q1.txt

Q2. Consider the following C code: (10 marks)

```
// write a MIPS instruction to initialize s0 to 6
```

$t0 = ((s0^3 - 93)^2 + s0) << 2$

$t1 = t0 / 4$

Q2.1: Write a MIPS program that performs the operation of the above C program.

Q2.2: What is the value of $t0 and $t1 after running your MIPS program (write your answer in a comment at the end of the code).

Submit your answer to Q2 in a file named A2_Q2.asm

Q3. What is the one C statement for the following MIPS code? assume we use integer division and the result from any multiplication fits into 32 bits. (5 marks)

```
add  $t0, $s3, $s3
add  $t0, $t0, $s2
addi $t1, $0, 5
mult $t0, $t1
mflo $t0
addi $t1, $s2, 6
div  $t0, $t1
mflo $t0
mult $t0, $t0
mflo $t0
sll  $t1, $s4, 2
and  $s0, $t0, $t1
```

Note: x^n can be calculated using the pow function: `pow(x, n)`

Submission Instructions: Compress all your files (including the assembly files) into one zip file and submit it to Canvas. You can resubmit an assignment, but the new submission overwrites the old one and receives a new timestamp.