CSI 3640: Computer Organization
Fall 2024, OAKLAND UNIVERSITY
MIPS Programming Assignment #1
Due: Tuesday, October 15, 2024, 11:59PM
100 points, amounts to 10% of the final grade.

In this first programming assignment, you will demonstrate your ability to implement programming control structures (sequential, selection and iteration) using the MIPS assembly language.

Write a MIPS assembly program that can draw a rectangle pattern onto the console output based on a given set of parameter values. To fully define the rectangle pattern, the following four parameters will be used:

1. ***Height***: the number of lines used to draw the rectangle
2. ***Width***: the number of characters in each line.
3. ***Border***: the character used to draw the rectangle's periphery
4. ***Fill***: the character used to draw the rectangle's inner points

For example, a rectangle of Height=5, Width=4, Border='*' and Fill ='@' would print as follows:

```
****
*@@*
*@@*
*@@*
****
```

You may assume that the Height and Width of the rectangle will always be positive integers. If either the height, or the width, or both equals to 1, then the rectangle should be drawn using the Border character only.

All parameter values of your assembly program will need to be defined and *hardcoded* in the data segment. The Length and Width of the rectangle should be stored as 32-bit words. Any parameter or constant of character type should be stored as bytes. For example, you could define a newline character in the data segment as:

```
newline: .byte '\n'
```

To load a single character from memory into a register, you could use the **`lb`** instruction instead of the **`lw`** instruction. The **`lb`** instruction (short for Load Byte) reads one byte of value from its argument address and loads the value into its argument register. For example, in the code segment, we can copy the above-mentioned newline character into the **`$a0`** register as follows:

```
la $s0, newline
lb $a0, 0($s0)
```

To get full credit for your work, each instruction of your MIPS code must be followed by a single-line comment explaining the purpose of that instruction.

Save your MIPS program in a single plaintext file and submit it to Moodle. Good Luck!