COA LAB ASSIGNMENT
RISC-V
(CS3P001)

Note: You can use this to test/run your code

1. Write a RISC-V assembly program to add 409932 + 409823. Given operands should be considered as immediate values. DO NOT USE ASSEMBLER DIRECTIVES. (Hint: Use **lui** instruction)

2. Write an assembly program to test if a given number N is prime or not. Save the Boolean result in a0.

3. Write a RISC-V assembly program to perform and analyze various logical operations on two given 8-bit values. The program should compute the results of AND, OR, XOR, NOT, as well as bit shifts, and store these results in memory.

4. Write an assembly program to find the number of ones in a 32-bit number(say N)

5. Write a RISC-V assembly program that calculates the sum of integers from 1 to N (where N is a positive integer). The value of N should be provided as input, and the result should be stored in memory.

6. In some cases, we can rotate an integer to the right by n positions (less than or equal to 31) so that we obtain the same number. For example: an 8-bit number 01010101 can be right rotated by 2, 4, or 6 places to obtain the same number. Write a RISC-V assembly program to efficiently count the number of ways we can rotate a number to the right such that the result is equal to the original number.

7. Write a RISC-V assembly program that calculates the factorial of a given positive integer N without using recursion. The result should be stored in a register.

8. Write a RISC-V assembly program to determine the number of even and odd elements in an array of integers. Store the counts of even and odd numbers in designated memory locations.

9. Suppose you decide to take your RISC-V device to some place with a high amount of radiation, which can cause some bits to flip, and consequently corrupt data. Hence, you decide to store a single bit checksum, which stores the parity of all the other bits, at the least significant position of the number (essentially you can now store only 31 bits of data in a register). Write a RISC-V assembly program, which adds two numbers taking care of the checksum. Assume that no bits flip while the program is running. Also assume that there is no overflow while adding two 31-bit numbers.

10. Write a RISC-V assembly program to perform the following operations on a null-terminated string:
    I. Count the Length: Calculate the number of characters in the string (excluding the null terminator).
    II. Convert to Uppercase: Convert all lowercase characters in the string to uppercase.