CPSC 3303 Programming Project 1

Instructions For this project, you must write three programs in the high-level language you learned in CPSC 1213. These exercises are **extensions** of the exercises assigned for the previous module. For the first two exercises, the extension consists of going from a unique character to a string of characters. For the third exercise, you must get "closer" to the computer and process the numbers as 0s and 1s using ONLY logic bitwise operations, rather than arithmetic operations.

For all these exercises, you cannot use built-in functions that perform these operations. In case of a doubt, check with your instructor.

Objectives of this project:

- To manipulate the variables as 0s and 1s using logic bitwise operation
- To practice conversion from a base (decimal, hexadecimal, or binary) to another base
- To distinguish between "numbers" and characters
-

What you need to do:

Programming Exercise 1 (18 points):

*This exercise is an extension of Programming Exercise 1 from the previous module's assignment. The extension consists of transitioning from a character to a **string** of characters.*
Write a program that prompts the user to enter a string ***s*** consisting of a 16-bit binary integer ***n***. Assuming that the 16-bit binary number ***n*** is unsigned, you must write a **function** that takes ***s*** as input and returns the equivalent number ***n***. Your main program must print out ***n*** in decimal.
Example: If the user enters the string "01000010", your program must print out 66.

Programming Exercise 2 (18 points):

This exercise is an extension of Programming Exercise 2 from the previous module's assignment.
Write a program that prompts the user to enter a string ***s*** consisting of a 4 digit hexadecimal number ***n***. Assuming that the number ***n*** is unsigned, you must write a **function** that takes ***s*** as input and returns the equivalent number ***n***. Your main program must print out *n* in decimal.
Example: If the user enters the string "02AE", your program must print out 686.

Programming Exercise 3 (24 points):

*This exercise is the **same** as the Programming Exercise 3 from the previous module, except that you are no longer allowed to use arithmetic operations such as `division` (/), `multiplication`, or `modulo` (%) to extract the bits. In this exercise use only logic bit-wise operations.*
Write a program that prompts the user to enter a positive integer ***n*** (0 up to 2^{32}-1). You must write a function that takes as input ***n*** and returns a **string *s*** representing the number ***n*** in binary. For this assignment, you **CANNOT use the arithmetic division by 2 or the modulo operation** to convert the number to binary. Your main program must print out ***s***.
Example: If the user enters the number 66, your program must print out 1000010.

Hints for Programming Exercise 3:
Hint 1: The number ***n*** is already in binary inside the memory. All you need is to "extract" or "read" the bits

individually. Read the next hints to know how.
Hint 2: Consider the number n=66. In the memory, 66 is 00000000010000**10**. I can isolate the least significant bit (**red** rightmost bit) by using the logic operation **AND** (**&**). Compute n & 1 = 66 & 1. Try the operation x & 1 with x taking different values to find out the effect: the operation "& 1" returns the value of the least significant bit!
Hint 3: Say, for example, you read the rightmost bit with the operation " & 1". How should you read the bit that is to the left of the least significant bit (i.e., the **blue** bit to the left of the red bit)? The hint is to *push* all the bits to the right **after** I extracted the rightmost bit. To push to the right, you can **shift right** (**>>**) the number n to the right.
n >> 1 = 66 >> 1 = 000000000010000**1**: all bits are pushed to the right. Now, that bit became the rightmost bit. . . And you know how to read the rightmost bit.

What you need to turn in:

- Electronic copies of your report (standalone) and source code (zipped) of your programs. All programming files (source code) must be put in a zipped folder named ***m2-name***, where *name* is your last name. Zip the folder and post it on Canvas. Submit separately (not inside the zipped folder) the report as a Microsoft Word or PDF file.
- Your report must:
    - State whether your code works.
    - Clearly explain how to compile and execute your code.
    - If needed/applicable, report/analyze (as appropriate) the results. The quality of analysis and writing is critical to your grade.
    - Good writing and presentation are expected.

How this assignment will be graded:

The program compiles and executes correctly without any apparent bugs. The code is well-designed and commented.	100% credit
The program compiles and executes with minor bugs. The code is well-designed and commented.	90% credit
The program compiles and executes with major bugs.	40% credit
The program compiles but does not produce any meaningful output.	5% credit

If the instructor needs additional communications/actions to compile and execute your code, then a 30% penalty will be applied. If the turn-in instructions are not correctly followed, 10 pts will be deducted.