

There are 1 questions for a total of 0 points.

- (0) 1. (*Binary arithmetic*): Under the hood of your computer, all information is expressed in binary, including numbers. So, operations on binary numbers (*i.e.*, *addition*, *subtraction*, *multiplication*, and *division*) are already implemented within your computer. However, the operations implemented in hardware are on binary numbers of a bounded size (such as 32 bits or 64 bits). Frequently, we need to support operations on large binary numbers that have thousands of bits rather than 32/64-bit (*for example, you may have heard about cryptographic operations in which you need to work with 1024-bit prime numbers*). In this programming assignment, we will work with large binary integers and understand the running time behaviour of various algorithms operating on binary numbers.

As you know by now, Python is an object-oriented programming Language, and one important feature of object-oriented design is **encapsulation**. Here, we can create program components where data can be bundled together with methods that act on the data, and the internal working of the component is private to the component. Since our goal in this assignment will be to do manipulations on large binary numbers, we will use encapsulation and define a Python class related to large binary numbers. Let us call this Python class **myBinaryInteger**. We will also use Python's **operator overloading and special methods** to enhance the utility of our class. The outline of the class and some of the methods to be implemented are given in the file `myBinaryInteger.py`. Your task in this assignment will be to complete the incomplete methods. For this assignment, you are asked to **self-evaluate** your methods, by cross-checking with the output of corresponding `int` operations and a Python built-in function for decimal-to-binary conversion called `bin()` and using `int(str, 2)` for binary-to-decimal conversion. Note that Python supports `int` operations for arbitrarily large integers. Some help code is given to you within `myBinaryInteger.py` to get you started.

SUBMISSION INSTRUCTIONS: All your program files should be in a directory `module_4_prog`. You will be asked to create a zip of this directory and submit this zip file. Your Python program should be in a file named `MyBinaryInteger.py`. Please note that **we do not have the autograder set** for this assignment. However, you are encouraged to submit your code. I will examine your code submission manually. Given this, please comment your code well so that it is readable.