Offline Assignment 3

Total Marks: **20**

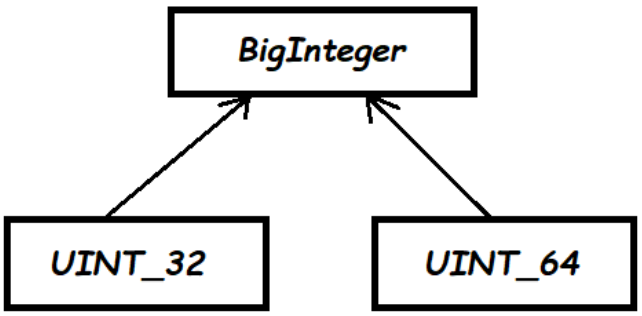
Problem Description

Suppose, you are working on a machine that supports *only* ***16-bit*** *integers* (e.g. *embedded processors* in various products). The *range* of values supported by this scheme is quite *small* ($[-2^{15}, 2^{15}-1]$ for *signed integers*, $[0, 2^{16}-1]$ for *unsigned integers*). However, you *may* need to work with *integers* from *larger ranges.*

In this assignment, we will generate provisions for large *unsigned integers* (*32-bits* and *64-bits*) using only *16-bit unsigned integers*.

Resources

- ★ We will be using ***C++*** as our *programming language* for this assignment.
- ★ To have access to ***16-bit*** *unsigned integers*, *include* the *header* ***cstdint*** in your program. You ***are highly encouraged*** to give the referred documentation a read.
- ★ To facilitate your coding, a template *header* ***BigInteger.h*** has been provided that contains the required *class definitions* and some *helper functions*. Download it from the aforementioned link. Carefully read the *commented* instructions and do your code on the *specified* places. Finally, *rename* the header file as instructed later and *include* it in your program.



N.B.:

- ★ Kindly keep an eye on the ***BigInteger.h*** *header* link for *minor revisions* to it in the future.
- ★ You ***can not*** change any of the *class definitions* or the *function signatures* that have been provided in it. However, you ***can*** add extra functions if you want to.
- ★ You ***can not*** use any **global** or **static** variables while solving this problem.
- ★ You ***can not*** work with *basic* ***int*** *data types*.
- ★ Do ***not*** assume the length of any ***data type***.
- ★ You ***can*** assume that the provided input will ***always*** be valid.

Explanation

- ➢ After including the ***cstdint*** *header* file in your program, you will have access to ***16-bit*** *unsigned integers*, namely ***uint16_t***.
- ➢ After including the ***BigInteger.h*** *header* file in your program, you will have access to the resources mentioned below. Carefully read the *comments* to understand what they are supposed to be.
  - For your convenience, ***typedef*** has been used to create an additional name (*alias*) ***UINT_16*** for ***uint16_t***. This is the *data type* that we will mostly be working with in this assignment.
  - A ***BigInteger*** *class*. It will be our **base** *class*. It contains the following (as stated in the *header* file):
    - One *member attribute*: a *string* ***INFO*** which stores some *information* of the *Integer*.
    - *Empty* as well as *parameterized constructors*.
    - Some *getters* and *setters*.
    - *Functions* that should be **overridden** in the **derived** *classes*:

```
void Set_bit_str(const char* bit_str);
const char* Generate_bit_str();
void ShiftLeft(UINT_16 k);
void ShiftRight(UINT_16 k);
void RotateClockwise(UINT_16 k);
void RotateAntiClockwise(UINT_16 k);
void Add(UINT_16 n);
void Increment();
void Print_bit_str();
```

- You ***can not*** add any extra *function(s)* here.

- An ***UINT_32*** *class*. It **inherits** the ***BigInteger*** *class*. It models ***32-bit*** *unsigned integers*. It must contain the following (as stated in the *header* file):
  - Necessary *member attributes* to store a ***32-bit*** *unsigned integer* using only ***16-bit*** *unsigned integers*. As included in the *header*, two ***16-bit*** *unsigned integer* variables will *suffice* in this case.
  - Some *parameterized constructors*.
  - Some *getters* and *setters*.
  - One *member function*: `void Add(const UINT_32& n)`
  - *Functions* **inherited** from the **base** ***BigInteger*** *class* that should be **overridden**.
  - You ***can*** add extra functions if you want to.
  - **BONUS:** Some *operators* that may be **overloaded**:

```
/// shift left (<<)
/// shift right (>>)
/// postfix increment (e.g. n++)
/// prefix increment (e.g. ++n)
/// addition (+) with an UINT_16
/// addition (+) with another UINT_32
/// casting operator uint32_t()
```

- Similarly, an ***UINT_64*** *class*. It **inherits** the ***BigInteger*** *class*. It models ***64-bit*** *unsigned integers*. It must contain the following (as stated in the *header* file):
  - Necessary *member attributes* to store a ***64-bit*** *unsigned integer* using only ***16-bit*** *unsigned integers*. Follow the declarations in ***UINT_32*** for reference.
  - Some *parameterized constructors*.
  - Some *getters* and *setters*. Follow the signatures in ***UINT_32*** for reference.
  - One *member function*: `void Add(const UINT_64& n)`
  - *Functions* **inherited** from the **base** ***BigInteger*** *class* that should be **overridden**.
  - You ***can*** add extra functions if you want to.
  - **BONUS:** Some *operators* that may be **overloaded**:

```
/// shift left (<<)
```

```
/// shift right (>>)
/// postfix increment (e.g. n++)
/// prefix increment (e.g. ++n)
/// addition (+) with an UINT_16
/// addition (+) with another UINT_64
/// casting operator uint64_t()
```

- Some *helper functions*. You ***are encouraged*** to add similar extra functions for your convenience.

Tasks

- ➢ *Include* the ***cstdint*** *header* file in your program.
- ➢ *Include* the ***BigInteger.h*** *header* file in your program.
- ➢ Complete the ***UINT_32*** *class* by providing *definitions* for the specified *functions*.
- ➢ Complete the ***UINT_64*** *class* by *declaring* the necessary *member attributes* and providing *definitions* for the specified *functions*.
- ➢ You ***are encouraged*** to attempt **Operator Overloading** for **bonus** marks.
- ➢ Write a *sample* ***C++*** program to demonstrate the usage of your *classes*.

Assignment Rules

- ★ Assignment must be submitted in **Moodle**. Submissions via email **will not** be accepted.
- ★ ***Follow*** the instructions regarding **submissions** mentioned below:

> **First, rename the file(s) containing your source code(s) as your *StudentId* (For example, if your student ID is *2006001*, the name of the *cpp* file should be *2006001.cpp* and the name of the *header* file should be *2006001.h*). After that, put your source code file(s) in a folder named as your *StudentId* i.e. *2006001*. Then, zip it. Finally, submit the zip file (*2006001.zip*) to Moodle. Any other file type will *not* be accepted. Make sure that your submitted file contains your source code(s).**
>
> ***Failure to follow these instructions will result in penalties.**

- ★ **Deadline** for the assignment is ***14/04/2022*** at ***11:55 PM***.
- ★ Avoid ***plagiarism*** with utmost priority, i.e., write all programs on your own. **DO NOT COPY** codes, programs or ideas from others and do not share your programs, ideas or codes with others. We regularly use copy checkers, so your submitted assignment will be checked for plagiarism against *your classmates* as well as

against the *internet*.

★ If any ***plagiarism*** gets detected:

- ➢ First time copier and copyee will receive **negative marking** (***-100%***) because of dishonesty. Their default is greater than those who will not submit. So, be **CAUTIOUS**.
- ➢ Repeated occurrences will lead to **severe** departmental action and that could jeopardize your academic career (you may be **EXPELLED** for up to two years as per the policy of **BUET**). We expect fairness and honesty from you. *Do not* disappoint us!

★ **No** request for extending the assignment deadline will be entertained.