CMPR121 - EXExam3

1. A rational number is a number that is expressed as the ratio of two integers, where the denominator should not be equal to zero.

Design a class named **Rational** that has the following private member variables:

* + Numerator – an int
  + Denominator – an int

In additional the class should have the following constructors, member functions, and overloading operators:

* A default constructor that set the numerator to 0 and the denominator to 1.
* A constructor that accepts the values of the numerator and denominator. **It must throw an exception if the denominator is zero.**
* A private **Normalize()** function that performs:
  + If the denominator is negative, then the negative sign is moved to the numerator. Examples: 1/-5 = -1/5 or -3/-4 = 3/4.
  + simplify if there is a GCD (greatest common divisor).

Examples: 2/4 = 1/2 or -6/3 = -2/1.

* Mutators (set) that changes the values of the numerator and denominator. **It must throw an exception if the denominator is zero.**
* Accessors (get) that retrieve the values of the numerator and denominator.
* An overload operator **==** that compares two Rational numbers are equal.
* An overload operator **<** that compares two Rational numbers; one is lesser than another.
* An overload operator **<<** that returns a string expression of the numerator/denominator. Examples: “1/2”and “5/9”.

**Note: MUST test all member functions thoroughly.**

1. Create program that has the following menu options:
   * **Vector container**
   * **List container**
   * **Stack container adaptor**
   * **Queue container adaptor**

Note: Use the given STL (standard template library) with the Rational numbers from Step #1 as the elements.

1. **Vector container** menu option displays the current element(s) from the vector and has the following sub-menu options:
   * Add (push\_back) an element – prompts and validates for a Rational number to be added to the back of the vector.
   * Insert an element at index – prompts and validates for a Rational number to be inserted to a valid index of the vector.
   * Retrieve an element from index – prompts for a valid index from the vector and retrieve the Rational number.

**Your program must throw an exception if te index is out-range.**

* + Erase element(s) – prompts and validates for a Rational number to be removed from the vector.
  + Sort the elements of the vector in ascending order.
  + Clear all elements from the vector.

**Note: You MUST use recursion to display all available elements from the vector.**

1. **List container** menu option displays the current element(s) from list and has the following sub-menu options:
   * Add (push\_front or push\_back) an element– prompts and validates for a Rational number to be added to the front or back of the list.
   * Insert an element after – prompts and validates for a Rational number to be inserted; prompts and validates an existing Rational number to be found for insertion.
   * Find an element – prompts and validates for a Rational number to be searched from the list.
   * Remove element(s) – prompts and validates for a Rational number to be removed from the list.
   * Sort the elements of the list in ascending order
   * Clear all elements from the list.

**Note: You MUST use recursion to display all available elements from the list.**

1. **Stack container adaptor** menu option displays the current element(s) from stack and has the following sub-menu options:
   * Push – prompt and validate for a Rational number, and push onto the stack.
   * Top – fetch and display the top element of the stack.
   * Pop – remove the element from the stack.

**Note: You MUST use recursion to display all available elements from the stack.**

1. **Queue container adaptor** menu option displays the current element(s) from queue and has the following sub-menu options:
   * Enqueue (push) – prompt and validate for a Rational number, and add the element to the queue.
   * Back – fetch and display the last element from the queue.
   * Front – fetch and display the first element from the queue.
   * Dequeue (pop) – remove the element from the queue.

**Note: You MUST use recursion to display all available elements from the queue.**