Lab 15 - Sets

New Attempt

Due Dec 5 by 11:59pm **Points** 20 **Submitting** a file upload
Available until Dec 16 at 11:59pm

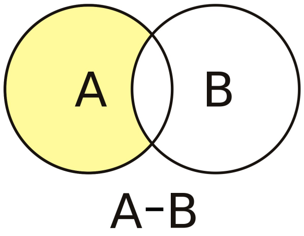
Advanced C++ Programming

Module 15 – Sets

The "Minus" Set Operation

(20 points)

Perform this lab individually



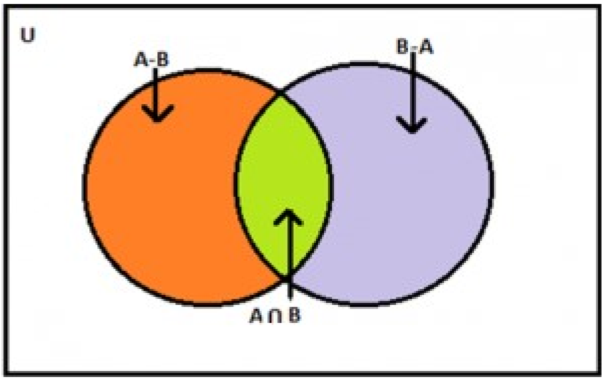
Summary

Enhance a generic **Set** class introduced in this Module.

Project 1: The “minus” operation on sets

In the lectures on linked lists and sets, we saw code which performed the *intersection* of two sets **A** and **B** (written **A ∩ B**) and the *union* of two sets (written **A U B**). Both of these operations return a new set (i.e., they do not change the original sets **A** or **B**).

Another common relationship is the *difference* between two sets **A** and **B** (written as **A – B**) called **A** minus **B**. In this operation, a new set is created which contains each element in set **A** unless that element is also in set **B**.



Looking at the picture above, two interesting relationships exist:

$$(A - B) \cup (A \cap B) = A$$

$$(A - B) \cup (A \cap B) \cup (B - A) = A \cup B$$

In this project, enhance the SetLinkedList.h (https://miracosta.instructure.com/courses/31330/files/7025702?wrap=1) (https://miracosta.instructure.com/courses/31330/files/7025702/download?download_frd=1) class (also located at the bottom of this lab and in the "Demonstration Programs" section of this Module) in three ways:

1. Create a new template function named **minus** with the following heading:

   ```
   template <typename T>
   Set<T> minus(Set<T> other_set)
   ```

   It should perform the *difference* operation described above, where the new set returned from the function contains all of the elements in calling set minus any element also in **other_set**.

2. Create an **equals** template function which returns **true** if all of the elements in the calling set are contained in the set passed as a parameter. The function should have a heading

   ```
   template <typename T>
   bool equals(Set<T> other_set)
   ```

   (Note: the **equals** function should also ensure that all of the elements in the set passed as a parameter are contained in the calling set.)

3. Create a **clear** template function which removes all of the elements in the calling set. The function should have a heading

   ```
   template <typename T>
   void clear()
   ```

   After running this function, the calling set should be empty (the **head** pointer equal to **nullptr**).

In a different program file containing **main**, test your new **minus**, **equals**, and **clear** functions with the following four test cases:

Test case 1: calling set = {“C”, “G”, “E”, “A”} and other(parameter) set = {“E”, “C”, “F”}

Test case 2: calling set = {“Carlos”, “John”, “Alice”} and other set = {“John”, “Henry”, “Maria”}

Test case 3: calling set = {5, 1, 3} and other set = {1, 3, 5, 7, 9}

Test case 4: calling set = {5} and other set = { }

Use the same sets for test cases 1 and 2, and for test cases 3 and 4. Use the **clear** function between test cases.

For each test case, print the contents of 5 sets: **A** (the calling set), **B** (other set), **A ∩ B**, **A – B**, and **(A – B) U (A ∩ B)**. Notice that the last set **(A – B) U (A ∩ B),** should have the same contents as set **A**. Use the **equals** function to demonstrate this point.

To do this, write a template function named **runTests** which takes two sets (**A** and **B**) and prints the contents of the 5 sets listed above plus the results of comparing set **A** with **(A – B) U (A ∩ B)** using the **equals** function. Then in **main** for each test case, create the calling set, create the "other" set, then call the **runTests** function. This template function should have as a heading:

```
template <typename T>
void runTests(Set<T> A, Set<T> B)
```

As an example, the output for the first test case might look like the following (use the little "n" to represent intersection):

```
Test case #1:
  Set A: A C E G
  Set B: C E F
  A n B: C E
  A - B: A G
  (A - B) U (A n B): A C E G
  equal?: yes
```

Along with your program files containing **main** and your enhanced version of the **Set** class, submit a screen snip or snips for each test case showing the results of the 5 required sets and the test for equality.

Links

Additional Files and Programs

SetLinkedList.h (https://miracosta.instructure.com/courses/31330/files/7025702?wrap=1)

Next Lab

Lab 16 - Binary Trees (https://miracosta.instructure.com/courses/31330/assignments/842814)

(https://miracosta.instructure.com/courses/31330/files/7025702/download?download_frd=1)	
Homework Assignment	**Prior Lab**
none	**Lab 14 - Linked Lists (https://miracosta.instructure.com/courses/31330/assignments/842812)**