

HW 8

In this homework we will write a *Set* class that represents sets of positive integers. You may not use a Set container to implement this. I implemented my version using an array of unsigned integers (called *slots*) to represent the set of integers. Thus, if 4 is a member of the set, then bit 4 (with the first bit being numbered zero) in the integer in `slots[0]` would be 1. If 33 was a member of the set, then bit 1 of the unsigned integer in `slots[1]` would be 1. If 34 is not a member of the set, then bit 2 of the unsigned integer in `slots[1]` would be 0. You can use another representation for your set, but you should not use a container.

Part A. Using member functions for all operators except for “<<”, implement the following:

A “+” *operator* that adds an integer to the set. If the set already contains the integer it is unchanged.

A “-” *operator* that removes an integer from the set. If the set does not contain the integer it is unchanged.

An “&” *operator* that “ands” the elements of a set, i.e. $s_3 = s_1 \& s_2$ means that element $e \in s_3$ iff $e \in s_1$ and $e \in s_2$.

A “~” *operator* that takes the inverse of a set. Thus, if $e \in s$, then $e \notin \sim s$. If $e \notin \sim s$, $e \in \sim s$.

A “/” *operator*. $e \in s_1 / s_2$ iff $e \in s_1$ and $e \notin s_2$, i.e., this is set difference.

A “<<” *operator* for printing out the elements of the set.

Implement a *copy constructor* and keep track of how many times it is called.

Part B. Using non-Member (free) functions, implement the operators above.

Parts A and B. The `main.cpp` file should work with your class.