**Data Structure:**

u\_int64\_t \_bits;

We will use an unsigned 64-bit number to represent set.

Bit position: 63…0

If bit in position n is 1, n is in the set.

If bit in position n is 0, n is NOT in the set.

**Example**

Empty set: \_bits=0

{0 to 63}: \_bits=0xFFFFFFFFFFFFFFFFUL

{0,1}: \_bits=0x3UL

**Feature:**

**1 Add an element n to set**

Mark bit in position n of set to 1.

**2 Remove an element n from set**

Mark bit in position n of set to 0.

**3 Union**

Bitwise OR of the two sets.

**4 Intersection**

Bitwise AND of the two sets.

**5 Difference (seta - setb)**

seta & (NOT setb)

**6 Operator==**

True if bit representations are the same.

**7 increment**

Left rotate 1 bit. \_bits<<1 | \_bits>>63

**8 decrement**

Right rotate 1 bit. \_bits>>1 | \_bits<<63

**9 Complement**

Bitwise NOT of the set.

**10 conversion if(a)**

True if bit representation is not 0.

**11 conversion if(!a)**

True if bit representation is 0.