

int i = 15;

Set of prime numbers: {2, 3, 5, 7, 11, 13, 17}

Positive multiples of 3 that are less than 10: {3, 6, 9}

iset64

class iset64 {
}

1. Empty set a = { }
2. The set we are implementing will have numbers between 0 to 63 only
3. In sets it does not matter what order the elements are in
Example: {1,2,3,4} is the same set as {3,1,4,2}
4. Number of elements in the above set = 4
5. In our set minimum number of element is 0 - Empty set
maximum number of element is 63
and the elements will be between 0 to 63

int a;
int a = 0;

iset64 a;

0
1
2

63

$\text{int } b4\ a;$
 $a = a + 3;$
 $a = a + 5;$
 $a = a + 75;$

63

Set of prime numbers: {2, 3, 5, 7, 11, 13, 17}
 Positive multiples of 3 that are less than 10: {3, 6, 9}

iset64

1. Empty set $a = \{ \}$
2. The set we are implementing will have numbers between 0 to 63 only
3. In sets it does not matter what order the elements are in
Example: {1,2,3,4} is the same set as {3,1,4,2}
4. Number of elements in the above set = 4
5. In our set minimum number of element is 0 - Empty set
maximum number of element is 64
and the elements will be between 0 to 63

$\text{cout} \ll a \ll \text{endl}$

{2, 5, 8}

{2, 3, 5, 17} {5, 2,

0 to 63

{3
 {2, 5, 8}

Adding an element to set
 $a = \{1, 2\}$
 $a += 5 = \{1, 2, 5\}$
 $a += \{10, 63\} = \{1, 2, 5, 63\}$

$$a = \{1, 2\}$$

$$\{1, 2\} \cup \{5\} = \{1, 2, 5\}$$

$$a = t = 5;$$

$$\{10, 1, 2, 5, 63\}$$

Removing an element

$a = \{1, 6, 10\}$

$a -= 6 = \{1, 10\}$

2

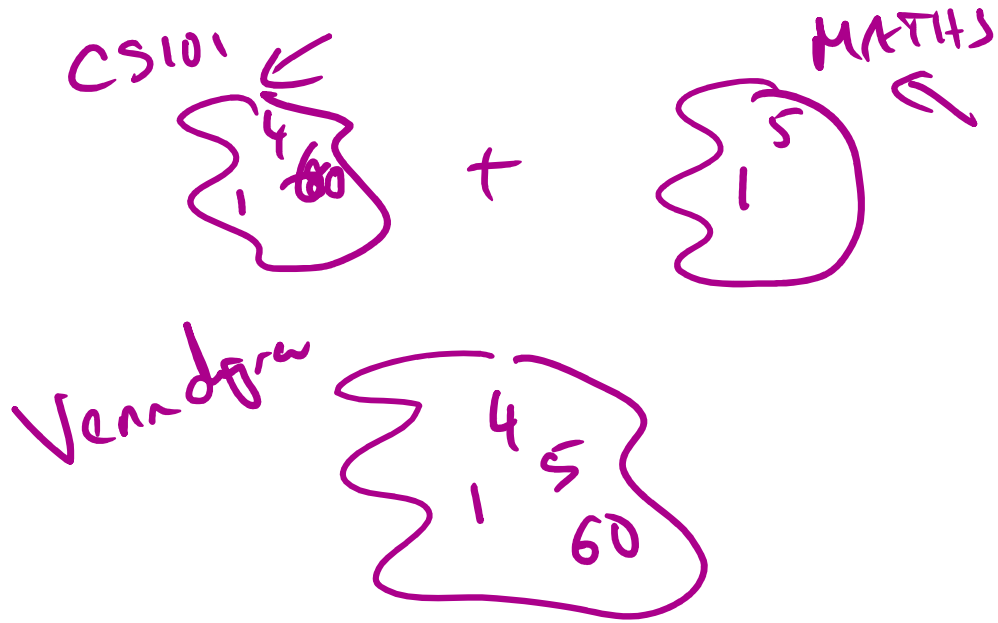
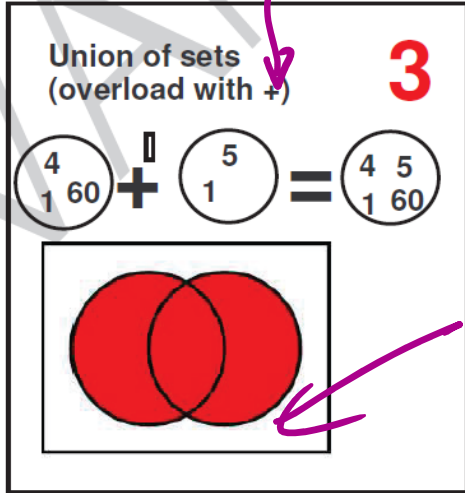
$a \leftarrow$

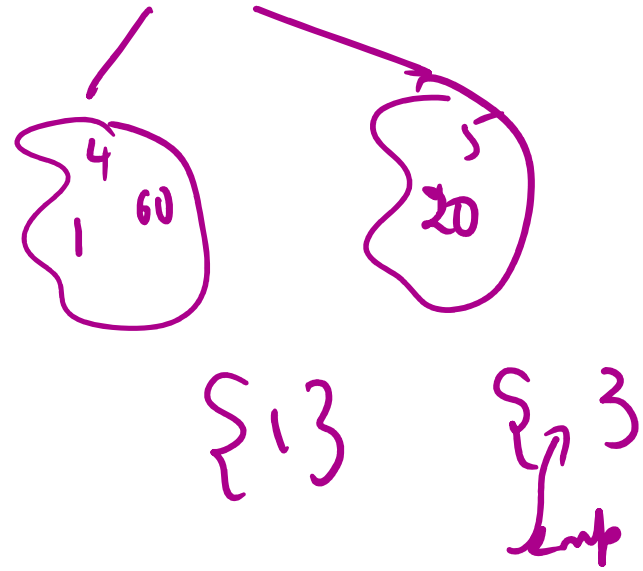
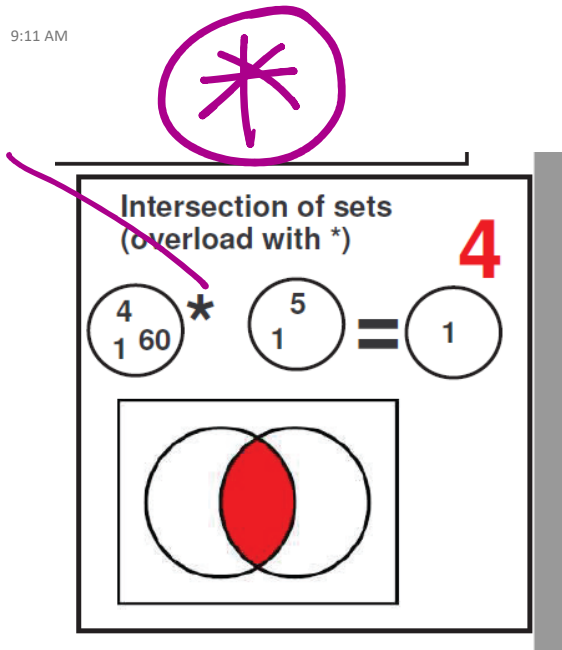
$a = a - 6$

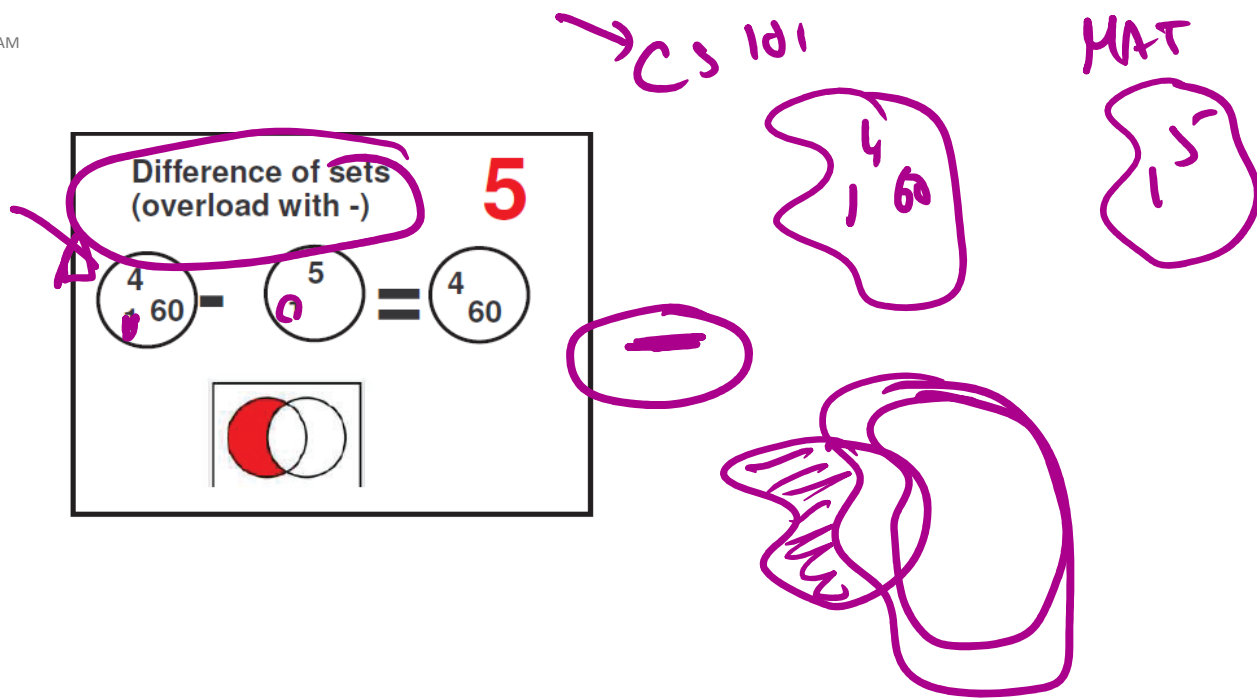
$a -= 6;$

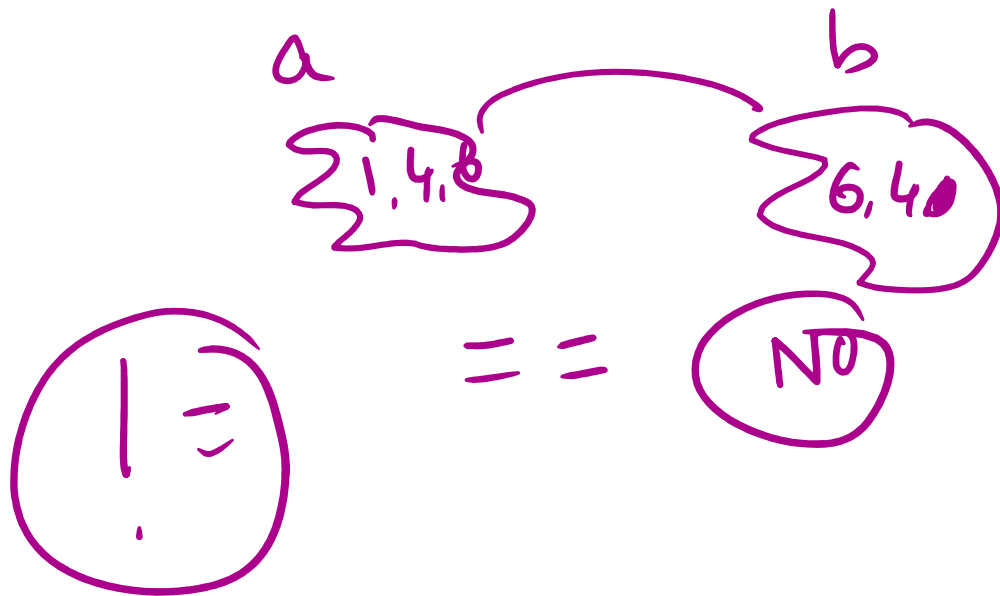
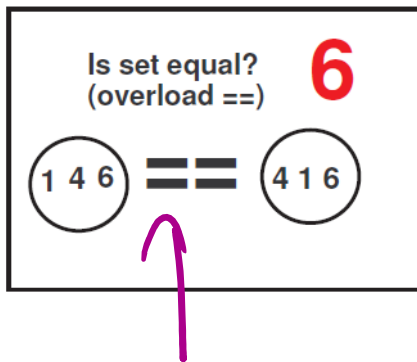
$\{1, 6, 10\}$

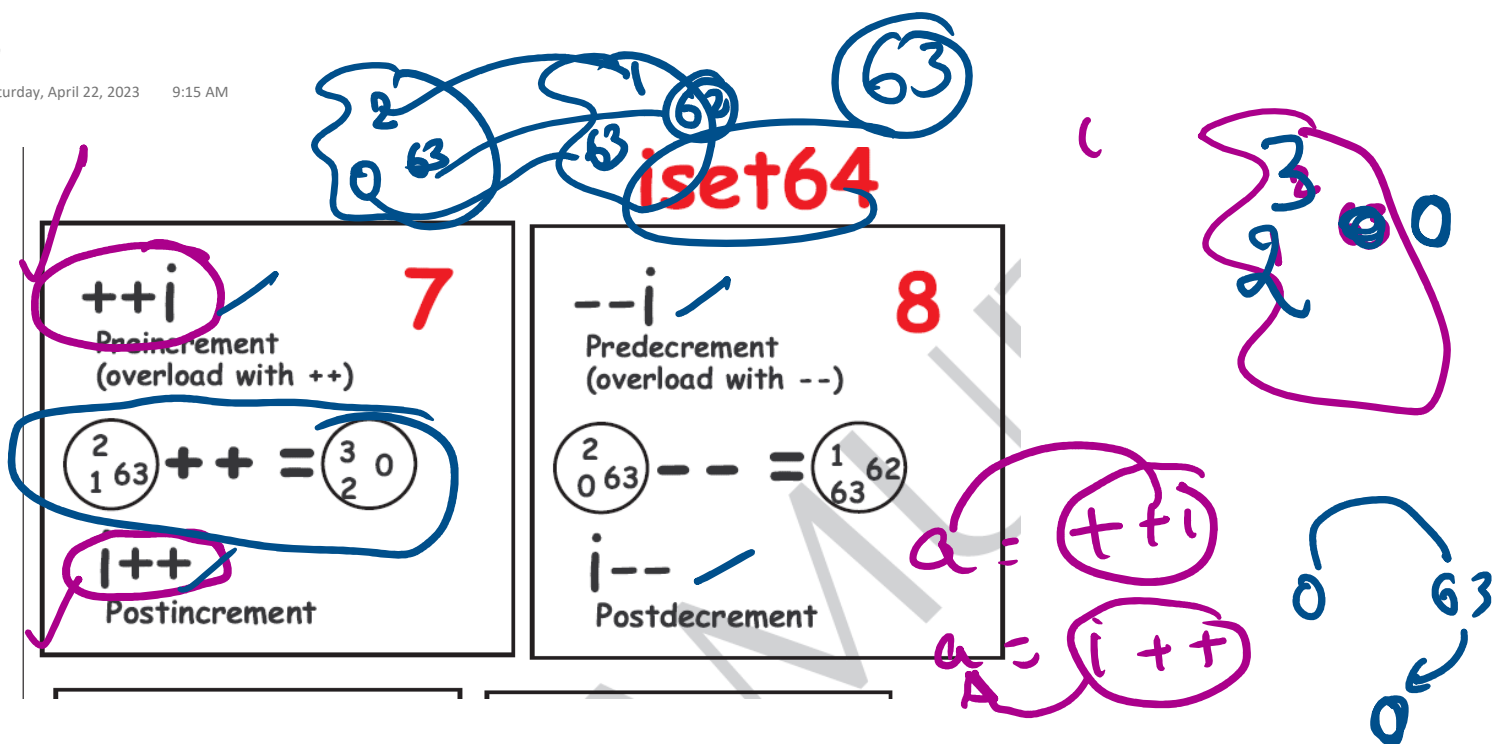
$\{1, 10\}$

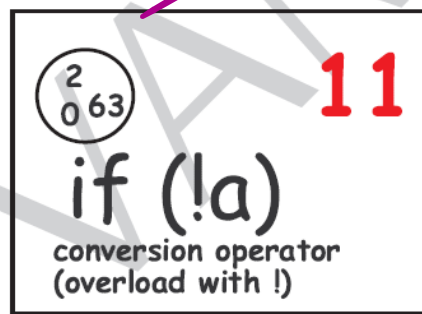
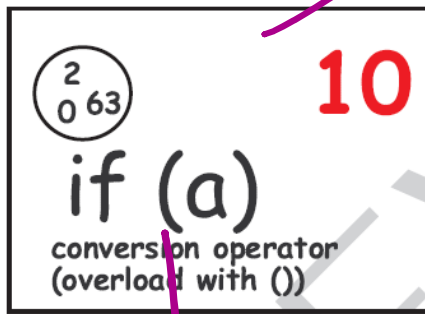








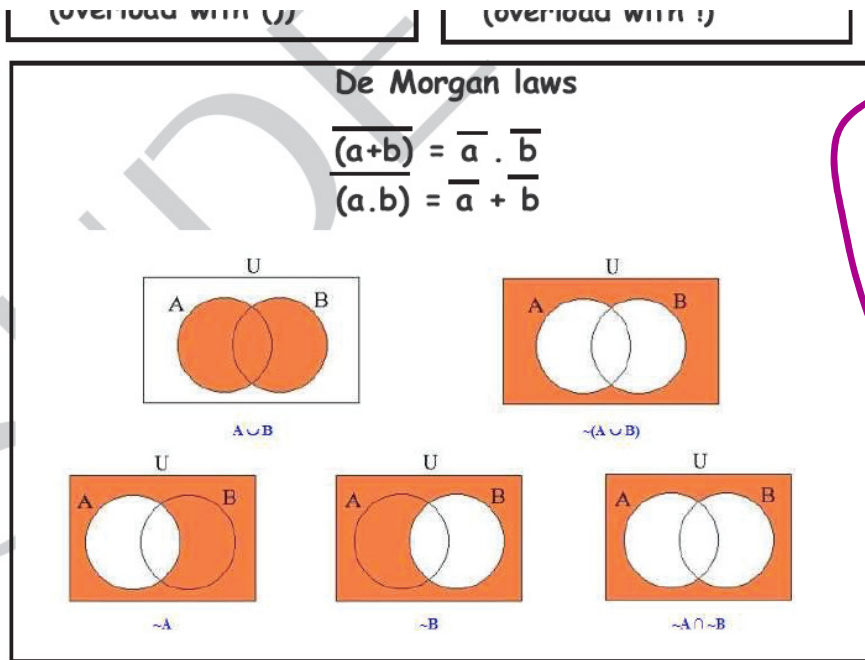




`if (a)`

`a = { 3`
`if (!a)`
`→`

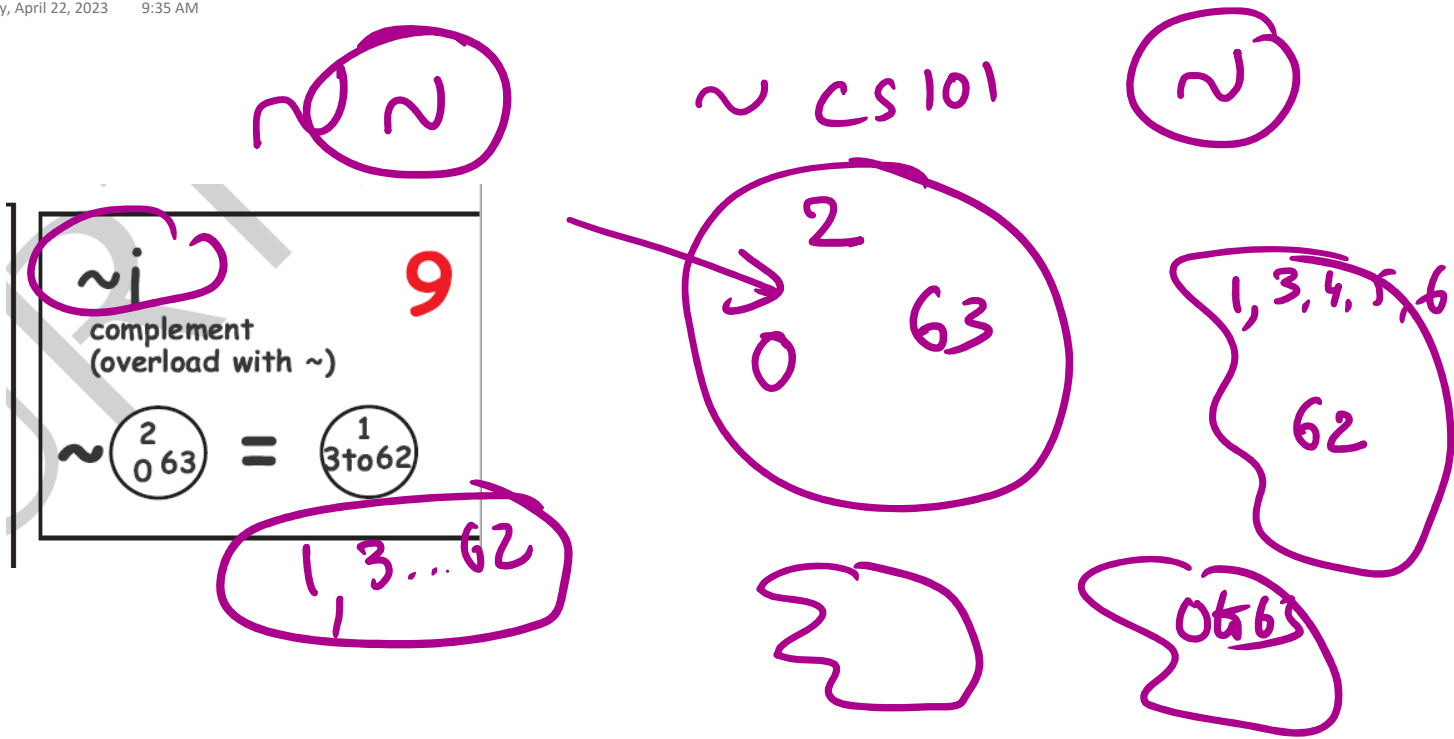
`int a = 0`
~~`int a = 25`~~
`if (a) {`
`{ →`
`3 else {`
`→`
`3`



$$\overline{a+b} = \overline{a} \cdot \overline{b}$$

$$\overline{a \cdot b} = \overline{a} + \overline{b}$$

TEST Bu



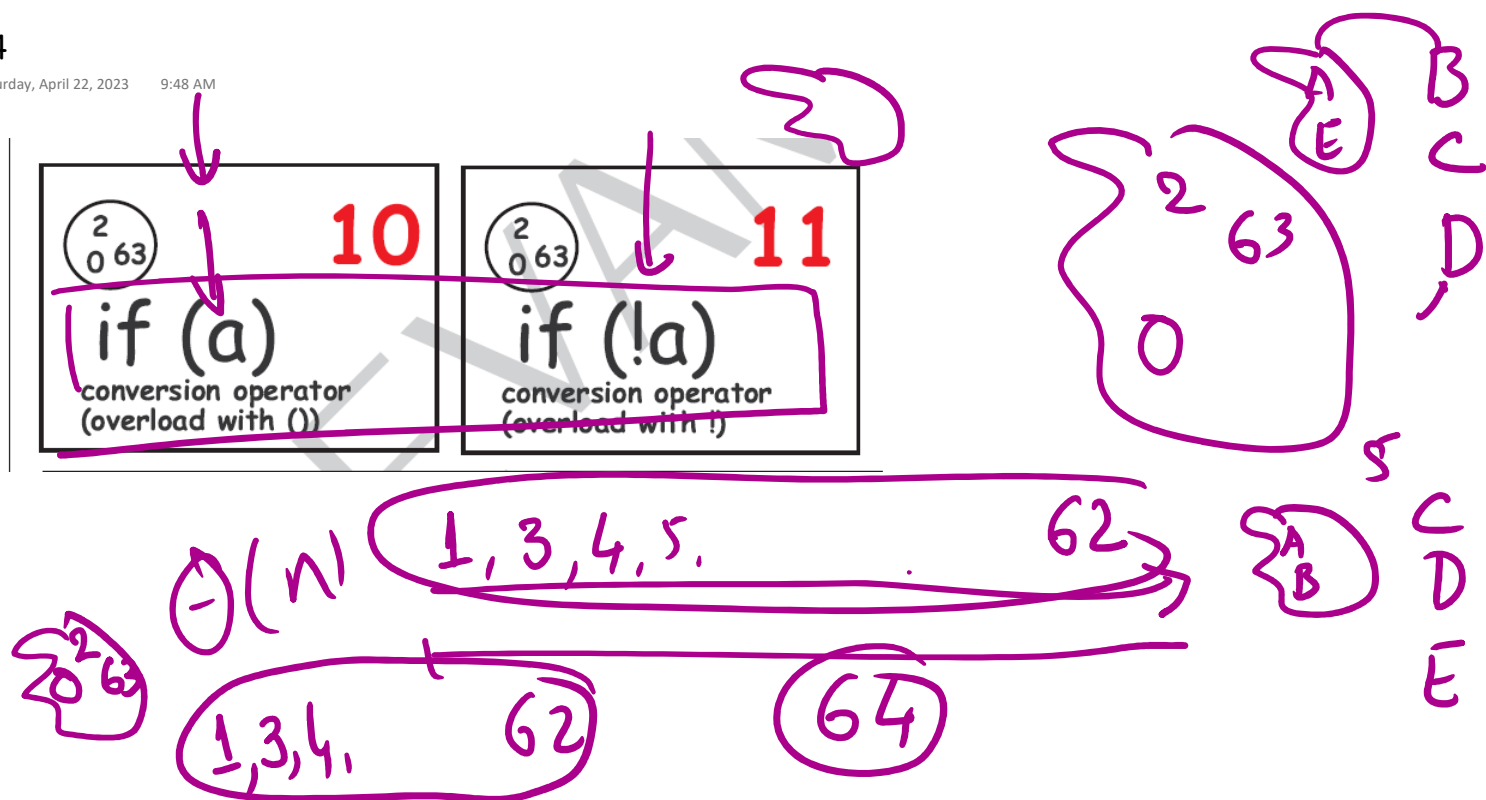
Handwritten notes and diagrams illustrating the code and its logic:

- Left side:** A vertical list of numbers: 2, 5, 61, 3, 2, 5, 61.
- Top center:** A circled note: `std::set set`.
- Top right:** A note: `int a[64] ← 10h`.
- Code snippet (lines 43-60):**

```

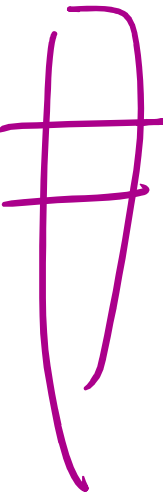
43
44
45 Declaration of iset64 class
46
47 class iset64{
48 public:
49     static const int N = 64;
50
51     static void set_display(bool b) {display = true; }
52 private:
53     //YOU cannot use
54     //int set [N];
55     //std::vector<int> set_;
56     static bool display;
57
58 };
59
60

```
- Diagram 1 (top right):** A horizontal array of 64 cells, indexed from 0 to 63. An arrow points from the `int a[64]` note to this diagram.
- Diagram 2 (bottom right):** A horizontal array of 64 cells, indexed from 0 to 63. An arrow points from the `std::set set` note to this diagram.



Alg

$N = 1000$
std::vector<int> a
int a[1000]
int a[64]
std::vector<int> a;



Must have 16 digits

4012888888881881

0 to 9

9 18

802216816616816828161

7 7 7 7 7 7 7

16 7

$0 < x < 18$

2 to 18

Multiples of 10

17

Must have 16 digits

4 0 1 2 8 8 8 8 8 8 8 8 1 8 8 1

4	0	1	2	8	8	8	8	8	8	8	8	1	8	8	1
x2	x2	x2	x2	x2	x2	x2	x2	x2	x2	x2	x2	x2	x2	x2	x2
8	2	16	16	16	16	16	16	16	16	16	16	2	16	16	16
8	2	7	7	7	7	7	7	7	7	7	7	2	7	7	7

8+0+2+2+7+8+7+8+7+8+7+8+7+8+2+8+7+1

= 90

This sum must be multiple of 10

ZERO MARKS GUARANTEED IF YOU USE STL string or char a[16]

YOUR OUTPUT MUST LOOK LIKE THIS.
do not worry about spaces, but easy to read

90

97

char GPT

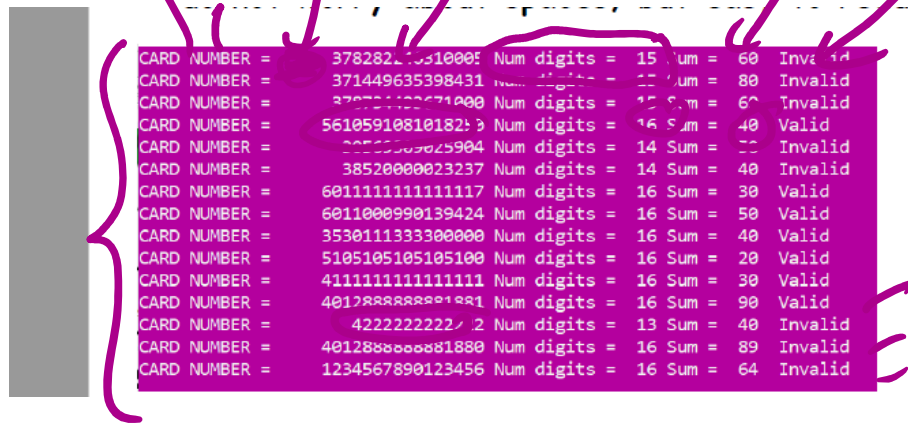
char a[16]

str::string

vec

std::vector

Frard



CARD NUMBER =	378282106310005	Num digits =	15	Sum =	60	Invalid
CARD NUMBER =	371449635398431	Num digits =	15	Sum =	80	Invalid
CARD NUMBER =	370701122671000	Num digits =	15	Sum =	60	Invalid
CARD NUMBER =	561059108101825	Num digits =	16	Sum =	40	Valid
CARD NUMBER =	00500505045904	Num digits =	14	Sum =	70	Invalid
CARD NUMBER =	38520000023237	Num digits =	14	Sum =	40	Invalid
CARD NUMBER =	601111111111117	Num digits =	16	Sum =	30	Valid
CARD NUMBER =	6011000990139424	Num digits =	16	Sum =	50	Valid
CARD NUMBER =	3530111333300000	Num digits =	16	Sum =	40	Valid
CARD NUMBER =	5105105105105100	Num digits =	16	Sum =	20	Valid
CARD NUMBER =	4111111111111111	Num digits =	16	Sum =	30	Valid
CARD NUMBER =	4012888888881881	Num digits =	16	Sum =	90	Valid
CARD NUMBER =	4222222222222	Num digits =	13	Sum =	40	Invalid
CARD NUMBER =	401288888881880	Num digits =	16	Sum =	89	Invalid
CARD NUMBER =	1234567890123456	Num digits =	16	Sum =	64	Invalid

```

/*-----
YOUR TEST MUST WORK FOR ALL DATA

```

```

378282246310005,
371449635398431,
378734493671000,
5610591081018250,
30569309025904,
38520000023237,
601111111111117,
6011000990139424,
3530111333300000,
5105105105105100,
411111111111111,
4012888888881881,
4222222222222,
4012888888881880,
1234567890123456,

```

{ }

Common place, -

for m

```

static void test() {
//NOTE YOU CANNOT USE
// "378282246310005" <----- THIS IS STRING
//MUST USE AS 378282246310005
}

```

→ check (~~378~~)
378

