Pointer

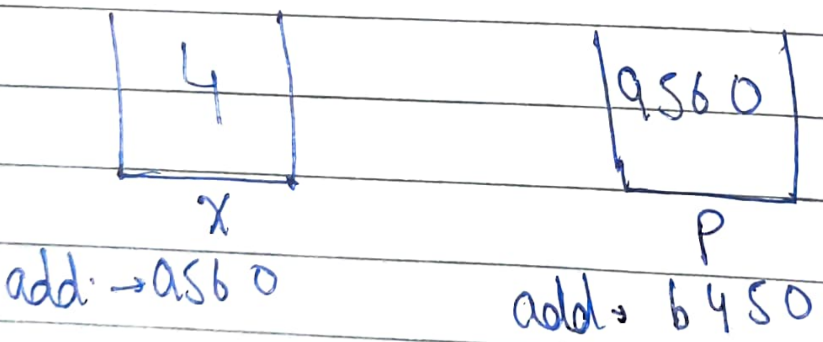
↳ Stores address of a variable

→ int x = 4

&x ⇝ It is the address of the variable x

⟹ int* p = &x;

↳ P is a pointer that stores address of x.



→ If we do → cout << p;

it will give us a560 as output.

Note the data type of variable has to be same as that of the pointer.

Date : / /
Page No.

⇒ declaration → data_type * pointer_name;

~~[illegible]~~

→
```
int x = 122;
int * p = &x;
Cout << *p;   // Output → 122
```
↳ This '*' is known as dereference operator

→ WAP to calculate sum of 2 numbers using pointers

```
int main () {
    int x, y;
    cin >> x >> y;
    int * p1 = &x;
    int * p2 = &y;
    Cout << *p1 + *p2;
}
```

Or

```
int main () {
    int x, y;
    int * p1 = &x;
    int * p2 = &y;
    cin >> *p1 >> *p2;
    Cout << *p1 + *p2;
}
```

Note → Syntax problem in pointers

```
int * p1 = &x;
int *  p2 = &y;
```
} ✓ This is right

```
int * p1 = &x, [illegible] p2 = &y
```
} ✗ This is wrong

→ Pointer Arithmatic

⇒ Increment and decrement

~~[illegible]~~

```
# int x=5;
  int * p = &x;
  (*p)++;  ——→ It gets incremented by 4 bytes
  Cout << P;  // prev. address + 4bytes
                 of x
```

→ bool/char → increment by 1

⇒ Null Pointer

⇒ int* ptr = NULL;

→ an address is reserved.

⇒ double pointer

```
  int x;
  int* ptr = &x ;
→ int** p = &ptr;
```

A double pointer is used to store the address of ~~of~~ a single pointer