Tutorial 51 - Pointers to Objects and Arrow Operator in C++

1. Pointer to Objects

- **Definition**: Pointers can store the address of an object, allowing access to its members.

Code Example 1: Pointer to Objects

```
#include<iostream>
using namespace std;
class Complex {
    int real, imaginary;
public:
    void setData(int a, int b) {
        real = a;
        imaginary = b;
    }
    void getData() {
        cout << "The real part is " << real << endl;
        cout << "The imaginary part is " << imaginary << endl;
    }
};
int main() {
    Complex *ptr = new Complex;
    (*ptr).setData(1, 54);
    (*ptr).getData();
    return 0;
}

```

- **Explanation**:
  a. Class `Complex` has private members `real` and `imaginary`.
  b. Member functions:
    - `setData(int a, int b)` assigns values to `real` and `imaginary`.
    - `getData()` prints these values.
  c. Object dynamically created using `new` and assigned to pointer `ptr`.
  d. Members accessed using dereference operator `(*ptr).`.

---

2. Arrow Operator (`->`)

- **Definition**: Simplifies access to members of a class when using a pointer to an object.

Code Example 2: Using the Arrow Operator

```
#include<iostream>
using namespace std;
class Complex {
    int real, imaginary;
public:
    void setData(int a, int b) {
        real = a;
        imaginary = b;
    }
    void getData() {
```

```
        cout << "The real part is " << real << endl;
        cout << "The imaginary part is " << imaginary << endl;
    }
};
int main() {
    Complex *ptr = new Complex;
    ptr->setData(1, 54);
    ptr->getData();
    // Array of objects
    Complex *ptr1 = new Complex[4];
    ptr1->setData(1, 4);
    ptr1->getData();
    return 0;
}

```

- **Explanation**:
  a. Arrow operator (`->`) simplifies access to members of an object through a pointer.
  b. For an object array, `ptr1->` accesses members of the first object.

Key Points

1. **Pointers to Objects**:
   - Pointers can store addresses of objects.
   - Use `(*ptr).member` to access members via pointers.
2. **Arrow Operator (`->`)**:
   - Short form of `(*ptr).member`.
   - Syntax: `ptr->member`.
   - Works for single objects and arrays.
3. **Object Array Access**:
   - `ptr1->` accesses members of the first object in an array of objects.

Notebook Short Notes

1. **Pointers to Objects**:
   - Store addresses of objects.
   - Access members using `(*ptr).member`.
   - Example:

```
Complex *ptr = new Complex;
(*ptr).setData(1, 54);
(*ptr).getData();

```

2. **Arrow Operator (`->`)**:
   - Simplifies access to members through pointers.
   - Equivalent to `(*ptr).member`.
   - Example:

```
ptr->setData(1, 54);
ptr->getData();

```

3. **Dynamic Object Array**:
   - Create with `new Complex[size];`.
   - Access members of the first object using `ptr1->`.
   - Example:

```
Complex *ptr1 = new Complex[4];
ptr1->setData(1, 4);
ptr1->getData();

```