# Tutorial 52 - Array of Objects Using Pointers in C++

### Definition:

An **array of objects** is an array where each element is an object of a class. Using pointers, we can dynamically allocate and manipulate arrays of objects.

---

### Code Example:

```
#include<iostream>
using namespace std;
class ShopItem {
    int id;
    float price;
public:
    void setData(int a, float b) {
        id = a;
        price = b;
    }
    void getData() {
        cout << "Code of this item is " << id << endl;
        cout << "Price of this item is " << price << endl;
    }
};
int main() {
    int size = 3;  // Array size
    ShopItem *ptr = new ShopItem[size]; // Dynamic array of objects
    ShopItem *ptrTemp = ptr; // Temporary pointer to iterate
    int p, i;
    float q;
    // Input loop
    for (i = 0; i < size; i++) {
        cout << "Enter Id and price of item " << i + 1 << endl;
        cin >> p >> q;
        ptr->setData(p, q); // Set data for the current object
        ptr++; // Move to the next object
    }
    // Output loop
    for (i = 0; i < size; i++) {
        cout << "Item number: " << i + 1 << endl;
        ptrTemp->getData(); // Get data for the current object
        ptrTemp++; // Move to the next object
    }
    return 0;
}

```

---

### Explanation:

1. **Class Structure**:
   - **Class Name**: `ShopItem`.
   - **Private Members**:

- `id` (int): Stores item ID.
- `price` (float): Stores item price.

- **Public Member Functions**:
  - `setData(int a, float b)`: Assigns values to `id` and `price`.
  - `getData()`: Prints `id` and `price`.

2. **Main Function**:
   - **Dynamic Array**:
     - Array of `ShopItem` objects is dynamically created using `new`.
     - Pointer `ptr` stores the base address of the array.
   - **Input Loop**:
     - Iterates over the array using pointer `ptr`.
     - Calls `setData()` to initialize objects.
     - Increments `ptr` to move to the next object.
   - **Output Loop**:
     - Iterates over the array using temporary pointer `ptrTemp`.
     - Calls `getData()` to print values of objects.
     - Increments `ptrTemp` to move to the next object.

---

### Key Points:

1. **Dynamic Array of Objects**:
   - Created using `new ClassName[size]`.
   - Pointer points to the base address of the array.
2. **Pointer Management**:
   - Use a temporary pointer (e.g., `ptrTemp`) to preserve the original pointer.
3. **Incrementing Pointer**:
   - Increment pointer in the input/output loop to access the next object in the array.

---

### Short Notes for Notebook:

1. **Array of Objects**:
   - Stores objects as elements.
   - Use pointers to dynamically allocate arrays.
2. **Code Snippet**:

```
ShopItem *ptr = new ShopItem[size];
ptr->setData(1, 100.5);
ptr->getData();

```

3. **Input Loop**:
   - Use `ptr->setData()` and increment `ptr`.
4. **Output Loop**:
   - Use a temporary pointer `ptrTemp` for `getData()`.
5. **Key Concept**:
   - Increment pointers to ensure different objects are accessed.