Tutorial 24 - Static Data Members & Methods in C++

Static Data Members

1. **Definition**:
   - Shared among all objects of the class.
   - Only one copy exists for all objects.
   - Default value is `0`.
2. **Key Points**:
   - Non-static data members have individual copies for each object.
   - Static data members are initialized outside the class using the scope resolution operator.

Static Methods

1. **Definition**:
   - Independent of any object.
   - Can only access static data members or call other static methods.
   - Called using the class name and the scope resolution operator (`ClassName::MethodName`).
2. **Key Points**:
   - Cannot access non-static members.
   - Useful for operations that are common across all objects.

Example: Employee Class

```
class Employee {
    int id;               // Non-static data member
    static int count;     // Static data member
public:
    void setData(void);   // Sets ID and increments count
    void getData(void);   // Displays ID and count
    static void getCount(void); // Displays count
};

```

Function Definitions

1. **setData**:
   - Inputs `id` from the user.
   - Increments the `count` static variable.

```
void Employee::setData(void) {
    cout << "Enter the id" << endl;
    cin >> id;
    count++;
}

```

2. **getData**:

- Displays `id` and `count`.

```
void Employee::getData(void) {
    cout << "The id of this employee is " << id
         << " and this is employee number " << count << endl;
}

```

3. **getCount**:
   - Displays the value of the static variable `count`.
   - Cannot access `id` (non-static member).

```
void Employee::getCount(void) {
    cout << "The value of count is " << count << endl;
}

```

4. **Static Member Initialization**:
   - Initialize the static member outside the class.

```
int Employee::count = 0;

```

Main Function

Demonstrates the usage of static data members and methods.

```
int main() {
    Employee harry, rohan, lovish;
    harry.setData();       // Set data for harry
    harry.getData();       // Display data for harry
    Employee::getCount();  // Call static method to display count
    rohan.setData();       // Set data for rohan
    rohan.getData();       // Display data for rohan
    Employee::getCount();  // Call static method to display count
    lovish.setData();      // Set data for lovish
    lovish.getData();      // Display data for lovish
    Employee::getCount();  // Call static method to display count
    return 0;
}

```

Program Output

For input:

- **harry**: ID = 1
- **rohan**: ID = 2
- **lovish**: ID = 3

The output is:

```
Enter the id
1
The id of this employee is 1 and this is employee number 1
The value of count is 1
```

```
Enter the id
2
The id of this employee is 2 and this is employee number 2
The value of count is 2
Enter the id
3
The id of this employee is 3 and this is employee number 3
The value of count is 3

```

Short Notes

1. **Static Data Members**:
   - Shared across all objects.
   - Only one copy exists.
   - Initialize outside the class: `int ClassName::memberName = value;`.
2. **Static Methods**:
   - Independent of objects.
   - Access only static members.
   - Called using: `ClassName::MethodName();`.
3. **Example**:
   - **Variables**:
     - `id`: Non-static, unique for each object.
     - `count`: Static, shared by all objects.
   - **Functions**:
     - `setData`: Inputs `id`, increments `count`.
     - `getData`: Displays `id` and `count`.
     - `getCount`: Displays `count`.
4. **Syntax Highlights**:
   - Static members: `static data_type memberName;`.
   - Static methods: `static returnType methodName();`.