Absolutely, Talha! Let’s break down the concepts you'll need for your OOP interview preparation, focusing on constructors, static variables, and static functions in C++.

### \*\*Order of Constructors\*\*

1. \*\*Default Constructor:\*\* A constructor that takes no arguments.

2. \*\*Parameterized Constructor:\*\* A constructor that takes arguments to initialize the object with specific values.

3. \*\*Copy Constructor:\*\* A constructor that initializes an object using another object of the same class.

#### \*\*Examples:\*\*

\*\*1. Default Constructor:\*\*

```cpp

class MyClass {

public:

MyClass() {

std::cout << "Default Constructor called\n";

}

};

int main() {

MyClass obj; // Default Constructor

return 0;

}

```

\*\*2. Parameterized Constructor:\*\*

```cpp

class MyClass {

public:

MyClass(int x) {

std::cout << "Parameterized Constructor called with value: " << x << "\n";

}

};

int main() {

MyClass obj(10); // Parameterized Constructor

return 0;

}

```

\*\*3. Copy Constructor:\*\*

```cpp

class MyClass {

public:

int value;

MyClass(int x) : value(x) {

std::cout << "Parameterized Constructor called\n";

}

MyClass(const MyClass& other) : value(other.value) {

std::cout << "Copy Constructor called\n";

}

};

int main() {

MyClass obj1(10); // Parameterized Constructor

MyClass obj2 = obj1; // Copy Constructor

return 0;

}

```

### \*\*Static Variables and Functions\*\*

\*\*Static Variables:\*\*

- Belong to the class rather than any instance.

- Shared among all instances of the class.

- Initialized only once.

\*\*Static Functions:\*\*

- Can be called without creating an instance of the class.

- Can only access static variables and other static functions.

#### \*\*Examples:\*\*

\*\*Static Variables:\*\*

```cpp

class Counter {

public:

static int count;

Counter() {

count++;

}

static void showCount() {

std::cout << "Count: " << count << "\n";

}

};

// Definition of static variable outside the class

int Counter::count = 0;

int main() {

Counter c1, c2, c3;

Counter::showCount(); // Static function to access static variable

return 0;

}

```

\*\*Static Functions:\*\*

```cpp

class MathUtils {

public:

static int add(int a, int b) {

return a + b;

}

};

int main() {

int result = MathUtils::add(5, 3); // Calling static function without an instance

std::cout << "Sum: " << result << "\n";

return 0;

}

```

### \*\*Order of Execution in Object Creation\*\*

When creating an object, the order of constructor calls is:

1. \*\*Static Members\*\* are initialized first (before any object is created).

2. \*\*Base Class Constructors\*\* are called before derived class constructors.

3. \*\*Member Initializers\*\* of the current class are initialized.

4. \*\*Constructor Body\*\* of the current class is executed.

#### \*\*Example:\*\*

```cpp

class Base {

public:

Base() {

std::cout << "Base Constructor\n";

}

};

class Derived : public Base {

public:

Derived() {

std::cout << "Derived Constructor\n";

}

};

int main() {

Derived d; // Output: Base Constructor\nDerived Constructor

return 0;

}

```

### \*\*Summary for Your Interview:\*\*

- \*\*Constructors:\*\* Know the default, parameterized, and copy constructors and their usage.

- \*\*Static Variables:\*\* Understand they belong to the class and are shared among all instances.

- \*\*Static Functions:\*\* Know they don’t require an object to be called and can only access static members.

Feel free to ask if you need more details on any of these topics!