Keywords in C++:

Keywords are reserved words in C++ with predefined meanings. These words form the building blocks of the language and cannot be used as identifiers (variable names, function names, etc.). Each keyword has a specific role in the C++ programming language.

Below is a detailed list of C++ keywords with their uses:

---

1. Control Flow Keywords

These keywords control the flow of the program:

- **if**: Introduces a conditional branch.
    - Example:

```
if (x > 0) {
   cout << "Positive number";
}
```

- **else**: Provides an alternative branch for an if condition.
    - Example:

```
if (x > 0) {
   cout << "Positive number";
} else {
   cout << "Not positive";
}
```

- **switch**, **case**: Used for multi-way branching.
    - Example:

```
switch (choice) {
   case 1:
      cout << "Option 1";
      break;
   case 2:
      cout << "Option 2";
      break;
   default:
      cout << "Default option";
```

}

- **while**: Executes a block of code repeatedly as long as a condition is true.
    - Example:

```
while (x < 10) {
   cout << x;
   x++;
}
```

- **do**: Similar to while but guarantees at least one execution.
    - Example:

```
do {
   cout << x;
   x++;
} while (x < 10);
```

- **for**: Used for looping with initialization, condition, and increment.
    - Example:

```
for (int i = 0; i < 10; i++) {
   cout << i;
}
```

- **break**: Exits a loop or switch statement prematurely.
- **continue**: Skips the rest of the loop iteration and proceeds to the next.

---

2. Data Handling Keywords

These manage variables and data:

- **const**: Defines a constant value.
    - Example:

```
const int MAX = 100;
```

- **extern**: Declares a variable defined elsewhere.
    - Example:

```
extern int x;
```

- **mutable**: Allows modification of a member variable in const objects.

- **static**: Retains the value of a variable between function calls or makes a class member shared across all objects.
    - Example:

```
static int count = 0;
```

---

3. Class and Object Keywords

These keywords relate to object-oriented programming:

- **class**: Defines a class.
    - Example:

```
class Car {
public:
   string brand;
};
```

- **private**, **public**, **protected**: Define access specifiers for class members.
- **friend**: Allows a function or class to access private/protected members of another class.
- **this**: Refers to the current object instance.
    - Example:

```
class Example {
   int x;
public:
   void setX(int x) {
      this->x = x;
   }
};
```

- **new**, **delete**: For dynamic memory allocation and deallocation.
    - Example:

```
int* ptr = new int;
delete ptr;
```

- **virtual**: Enables runtime polymorphism in derived classes.
    - Example:

```
virtual void display();
```

---

4. Exception Handling Keywords

These manage program exceptions:

- **try**, **catch**, **throw**: Handle runtime errors.
    - Example:

```
try {
   throw "An error occurred!";
} catch (const char* msg) {
   cout << msg;
}
```

---

5. Namespace and Scope Keywords

These manage scope and avoid name conflicts:

- **namespace**: Groups related classes, functions, etc.
    - Example:

```
namespace MyNamespace {
   int x = 10;
}
```

- **using**: Allows usage of a namespace or aliasing.
    - Example:

```
using namespace std;
```

---

6. Memory Management Keywords

- **sizeof**: Returns the size of a data type or object.
    - Example:

```
cout << sizeof(int);
```

- **typeid**: Retrieves type information of a variable.
    - Example:

```
cout << typeid(x).name();
```

---

7. Program Structure Keywords

- **typedef**: Creates an alias for a type.
    - Example:

typedef unsigned int uint;

- **enum**: Defines an enumeration.
    - Example:

enum Days { MON, TUE, WED };

- **union**: Defines a union, which shares memory for its members.
    - Example:

union Data {
  int i;
  float f;
};

---

8. Operator Keywords

- **and**, **or**, **not**: Logical operators.
- **bitand**, **bitor**, **xor**, **compl**: Bitwise operators.

---

9. Special Purpose Keywords

- **goto**: Jumps to a labeled statement (use discouraged).
    - Example:

goto label;
label:
  cout << "Jumped!";

- **volatile**: Indicates that a variable can be changed outside the program's scope.

Key Points

1. **Reserved Words**: Keywords cannot be redefined or used for variable names.
2. **Case Sensitivity**: All keywords are case-sensitive.
3. **Evolution**: Some keywords were added in C++11, C++14, and later versions (e.g., constexpr, decltype).

By understanding the use and context of each keyword, you can write more structured and efficient C++ programs.