Initializer List in C++

Initializer list is used to initialize data members.

An initializer list starts after the constructor name and its parameters. The list begins with a colon (:) and is followed by the list of variables that are to be initialized – all of the variables are separated by a comma with their values in curly brackets.

Syntax

```
Constructorname(datatype value1, datatype value2):datamember(value1),datamember(value2)
{
    ...
}
```

With Default Constructor

main.cpp

```
#include<iostream>
using namespace std;
//CONSTRUCTOR MEMBER INITIALIZER LIST
class InitializerDemoWithDefaultConstructor
{
    private:
    int num1,num2,num3;
    public:

    InitializerDemoWithDefaultConstructor():num1(10),num2(20),num3(30) {}

    void disp()
    {
        cout<<" Num 1 "<<num1<<" Num2 "<<num2<<" Num3 "<<num3<<endl;
    }

};
int main(void)
{
    InitializerDemoWithDefaultConstructor t;
    t.disp();
    return 0;
}
```

input

```
Num 1 10 Num2 20 Num3 30
```

With Parameterize Constructor

main.cpp

```cpp
#include<iostream>
using namespace std;
//CONSTRUCTOR MEMBER INITIALIZER LIST
class InitializerDemoWithParameterizeConstructor
{
    private:
    int num1,num2,num3;
    public:

    InitializerDemoWithParameterizeConstructor(int x,int y,int z):num1(x),num2(y),num3(z)
    {
        this->num1=num1;
        this->num2=num2;
        this->num3=num3;
    }
    void disp()
    {
        cout<<" Num 1 "<<num1<<" Num2 "<<num2<<" Num3 "<<num3<<endl;
    }
};
int main(void)
{
    InitializerDemoWithParameterizeConstructor t(100,200,300);
    t.disp();
    return 0;
}
```

input

```
Num 1 100 Num2 200 Num3 300
```

What are initializer lists in C++?

1. Initializing `const` data member

`const` data members can be initialized only once, so they must be initialized in the initialization list.

main.cpp

```cpp
#include<iostream>
using namespace std;

class Base
{
  private:
  const int var;
  public:
  Base(int constant_value):var(constant_value)
  {
    cout << "Value is " << var;
  }
};

int main()
{
  Base myobject(10);
}
```

input

```
Value is 10
```

2. Initializing a reference type data member

An initialization list is used to initialize a data member of reference type. Reference types can only be initialized once.

main.cpp

```cpp
#include<iostream>
using namespace std;

class Base
{
  private:
  int &ref;
  public:
  Base(int &passed):ref(passed)
  {
    cout << "Value is " << ref;
  }
};

int main()
{
  int ref=10;
  Base myobject(ref);
  return 0;
}
```

input

```
Value is 10
```

3. Initializing member objects which do not have a default constructor

If you have a field that has no default constructor (or a parent class with no default constructor), then you must specify which constructor you wish to use.

main.cpp

```cpp
#include<iostream>
using namespace std;

class Base_
{
  public:
  Base_(int x)
  {
    cout << "Base Class Constructor. Value is: " << x << endl;
  }
};

class InitilizerList_:public Base_
{
  public:
  // default constructor using initializer list
  InitilizerList_():Base_(10)
  {
    cout << "InitilizerList_'s Constructor" << endl;
  }
};

int main()
{
  InitilizerList_ mylist;
  return 0;
}
```

```
Base Class Constructor. Value is: 10
InitilizerList_'s Constructor
```

base class initialize you can visit after inheritance tutorials