**C++ Templates**

Templates are powerful features of C++ which allows you to write generic programs. In simple terms, you can create a single function or a class to work with different data types using templates.

Templates are often used in larger codebase for the purpose of code reusability and flexibility of the programs.

The concept of templates can be used in two different ways:

* Function Templates
* Class Templates

### How to declare a function template?

A function template starts with the keyword **template** followed by template parameter/s inside  **< >** which is followed by function declaration.

**template** <**class** T>

T someFunction(T arg)

{

... .. ...

}

### Example 1: Function Template to find the largest number

**Program to display largest among two numbers using function templates.**

// If two characters are passed to function template, character with larger ASCII value is displayed.

#include <iostream>

using namespace std;

// template function

template <class T>

T Large(T n1, T n2)

{

return (n1 > n2) ? n1 : n2;

}

int main()

{

int i1, i2;

float f1, f2;

char c1, c2;

cout << "Enter two integers:\n";

cin >> i1 >> i2;

cout << Large(i1, i2) <<" is larger." << endl;

cout << "\nEnter two floating-point numbers:\n";

cin >> f1 >> f2;

cout << Large(f1, f2) <<" is larger." << endl;

cout << "\nEnter two characters:\n";

cin >> c1 >> c2;

cout << Large(c1, c2) << " has larger ASCII value.";

return 0;

}

**Output**

Enter two integers:

5

10

10 is larger.

Enter two floating-point numbers:

12.4

10.2

12.4 is larger.

Enter two characters:

z

Z

z has larger ASCII value.

### Example 2: Swap Data Using Function Templates

**Program to swap data using function templates.**

#include <iostream>

using namespace std;

template <typename T>

void Swap(T &n1, T &n2)

{

T temp;

temp = n1;

n1 = n2;

n2 = temp;

}

int main()

{

int i1 = 1, i2 = 2;

float f1 = 1.1, f2 = 2.2;

char c1 = 'a', c2 = 'b';

cout << "Before passing data to function template.\n";

cout << "i1 = " << i1 << "\ni2 = " << i2;

cout << "\nf1 = " << f1 << "\nf2 = " << f2;

cout << "\nc1 = " << c1 << "\nc2 = " << c2;

Swap(i1, i2);

Swap(f1, f2);

Swap(c1, c2);

cout << "\n\nAfter passing data to function template.\n";

cout << "i1 = " << i1 << "\ni2 = " << i2;

cout << "\nf1 = " << f1 << "\nf2 = " << f2;

cout << "\nc1 = " << c1 << "\nc2 = " << c2;

return 0;

}

**Output**

Before passing data to function template.

i1 = 1

i2 = 2

f1 = 1.1

f2 = 2.2

c1 = a

c2 = b

After passing data to function template.

i1 = 2

i2 = 1

f1 = 2.2

f2 = 1.1

c1 = b

c2 = a

### How to declare a class template?

**template** <**class** T>

class className

{

... .. ...

public:

T var;

T someOperation(T arg);

... .. ...

};

### How to create a class template object?

To create a class template object, you need to define the data type inside a < > when creation.

className<dataType> classObject;

For example:

className<int> classObject;

className<float> classObject;

className<string> classObject;

### Example 3: Simple calculator using Class template

Program to add, subtract, multiply and divide two numbers using class template

#include <iostream>

using namespace std;

template <class T>

class Calculator

{

private:

T num1, num2;

public:

Calculator(T n1, T n2)

{

num1 = n1;

num2 = n2;

}

void displayResult()

{

cout << "Numbers are: " << num1 << " and " << num2 << "." << endl;

cout << "Addition is: " << add() << endl;

cout << "Subtraction is: " << subtract() << endl;

cout << "Product is: " << multiply() << endl;

cout << "Division is: " << divide() << endl;

}

T add() { return num1 + num2; }

T subtract() { return num1 - num2; }

T multiply() { return num1 \* num2; }

T divide() { return num1 / num2; }

};

int main()

{

Calculator<int> intCalc(2, 1);

Calculator<float> floatCalc(2.4, 1.2);

cout << "Int results:" << endl;

intCalc.displayResult();

cout << endl << "Float results:" << endl;

floatCalc.displayResult();

return 0;

}

**Output**

Int results:

Numbers are: 2 and 1.

Addition is: 3

Subtraction is: 1

Product is: 2

Division is: 2

Float results:

Numbers are: 2.4 and 1.2.

Addition is: 3.6

Subtraction is: 1.2

Product is: 2.88

Division is: 2

### Class template with multiple parameters

|  |
| --- |
| #include<iostream>  using namespace std;    // Class template with two parameters  template<class T1, class T2>  class Test  {          T1 a;          T2 b;      public:          Test(T1 x, T2 y)          {              a = x;              b = y;          }          void show()          {              cout << a << " and " << b << endl;          }  };    // Main Function  int main()  {      // instantiation with float and int type      Test <float, int> test1 (1.23, 123);        // instantiation with float and char type      Test <int, char> test2 (100, 'W');        test1.show();      test2.show();        return 0;  } |

Output:

1.23 and 123

100 and W