Class Template in C++

A class template starts with the keyword template followed by template parameter(s) inside <> which is followed by the class declaration.

```
template <class T>

class className {

  private:
    T var;

    ... .. ...

   public:

   T functionName(T arg);
    ... .. ...

};
```

In the above declaration, T is the template argument which is a placeholder for the data type used, and class is a keyword.

Class Templates like function templates, class templates are useful when a class defines something that is independent of the data type. Can be useful for classes like LinkedList, BinaryTree, Stack, Queue, Array, etc.

Following is a simple example of a template Array class.

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

template <typename T> class Array {
private:
    T* ptr;
    int size;

public:
    Array(T arr[], int s);
    void print();
};

template <typename T> Array<T>::Array(T arr[], int s)
{
    ptr = new T[s];
    size = s;
    for (int i = 0; i < size; i++)
        ptr[i] = arr[i];
}

template <typename T> void Array<T>::print()
{
    for (int i = 0; i < size; i++)
        cout << " " << *(ptr + i);
    cout << endl;
}

int main()
{
    int arr[5] = { 1, 2, 3, 4, 5 };
    Array<int> a(arr, 5);
    a.print();
    return 0;
}
```

Output

```
 1 2 3 4 5
```