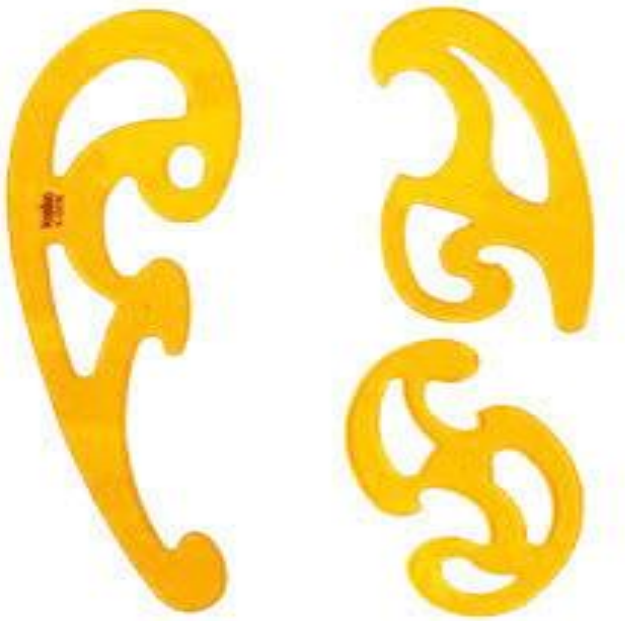
Template

CSE2013-17F

Template

- Template Ruler
  - The function is determined, but the color is not determined.



Template

```
int Add(int a, int b)
{
        return a+b;
}
```

```
template <typename T>
T Add(T a, T b)
{
        return a+b;
}
```

template <typename T> == template <class T>

Template

```
template <typename T>
T Add(T a, T b)
{
        return a+b;
}

...
        Add(10, 20);

        Add(1.1, 2.2);
...
```


```
int Add(int a, int b)
{
        return a+b;
}
```

```
double Add(double a, double b)
{
        return a+b;
}
```

Function Template

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

template <typename T>
T Add(T a, T b)
{
	return a+b;
}

int main()
{
	cout << Add(10, 20) << endl;
	cout << Add(1.1, 2.2) << endl;
	cout << Add<string>("S", "TR") << endl;
	return 0;
}
```

```
30
3.3
STR
```

Function Template

- Template for more than one data type

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

template <typename T>
void PrintData(T a, T b)
{
	cout << a << endl;
	cout << b << endl;
}

int main()
{
	PrintData(1, 2);
	PrintData(3, 2.5);    // error
	return 0;
}
```

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

template <typename T1, typename T2>
void PrintData(T1 a, T2 b)
{
	cout << a << endl;
	cout << b << endl;
}

int main()
{
	PrintData(1, 2);
	PrintData(3, 2.5);
	return 0;
}
```

Template Specialization

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

template <typename T1, typename T2>
void PrintData(T1 a, T2 b) {
	cout << a << endl;
	cout << b << endl;
}
template <>
void PrintData(char *a, int b) {
	for (int i = 0; i < b; i++)
		cout << a << endl;
}

int main() {
	char *str = "Good Morning";
	PrintData(1, 2);
	PrintData(3, 2.5);
	PrintData(str, 3);	// special case
}
```

```
1
2
3
2.5
Good Morning
Good Morning
Good Morning
```

Class Template

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

template <typename T>
class Data {
	T data;
public:
	Data(T d) {
		data = d;
	}
	void SetData(T d) {
		data = d;
	}
	T GetData() {
		return data;
	}
};

int main() {
	Data<int>  d1( 0 );
	d1.SetData(10);
	Data<char>  d2( 'a' );

	cout << d1.GetData() << endl;
	cout << d2.GetData() << endl;
}
```

```
10
a
```

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

template <typename T>
class Data {
	T data;
public:
	Data(T);
	void SetData(T);
	T GetData();
};

template <typename T>
Data<T>::Data(T d) {
	data = d;
}
template <typename T>
void Data<T>::SetData(T d) {
	 data = d;
}
template <typename T>
T Data<T>::GetData() {
	return data;
}

int main() {
	Data<int>  d1( 0 );
	d1.SetData(10);
	Data<char>  d2( 'a' );

	cout << d1.GetData() << endl;
	cout << d2.GetData() << endl;
}
```