Templates

- Template is simple and very powerful tool in c++.
- Templates are the foundation of generic programming ,which involves writing code in way that is independent in a way that is independent of any particular type.
- A template is a blueprint or formula for creating generic class or function.
- 2 types

  *Function template

  *class template

Function templates

- ➔ Function templates are special functions that can operate with generic types
- ➔ We write a generic function that can be used for different data types.

Function Overloading vs Function Template

- Function overloading –

```
int add(int x, int y){}
float add(float x, float y){}
double add(double x, double y){}

int main ()
{
    add(5,4);
    add(2.3f, 4.2f)
    add(5.3232, 42324.453)
}
```

- Function Template –

```
template <typename T>
T add(T x, T y)
{}

int main()
{
    add<int>(3, 7);
    add<float>(3.3, 7.5);
    add<double>(3.55, 7.66);
}
```

Class templates in C++

- ➔ Sometimes,you need a class implementation that is same for all classes,only the datatypes used are different.
- ➔ Normally,you would need to create a different class for each data type OR create different member variables and function with in single class .
- ➔ In class templates we write a class that can be used for different data types

Example


```
class Stack
{
    public:
    int arr[5]
    private:
    push();
    pop();
}
```


```
class Stack
{
    public:
    char arr[5]
    private:
    push();
    pop();
}
```