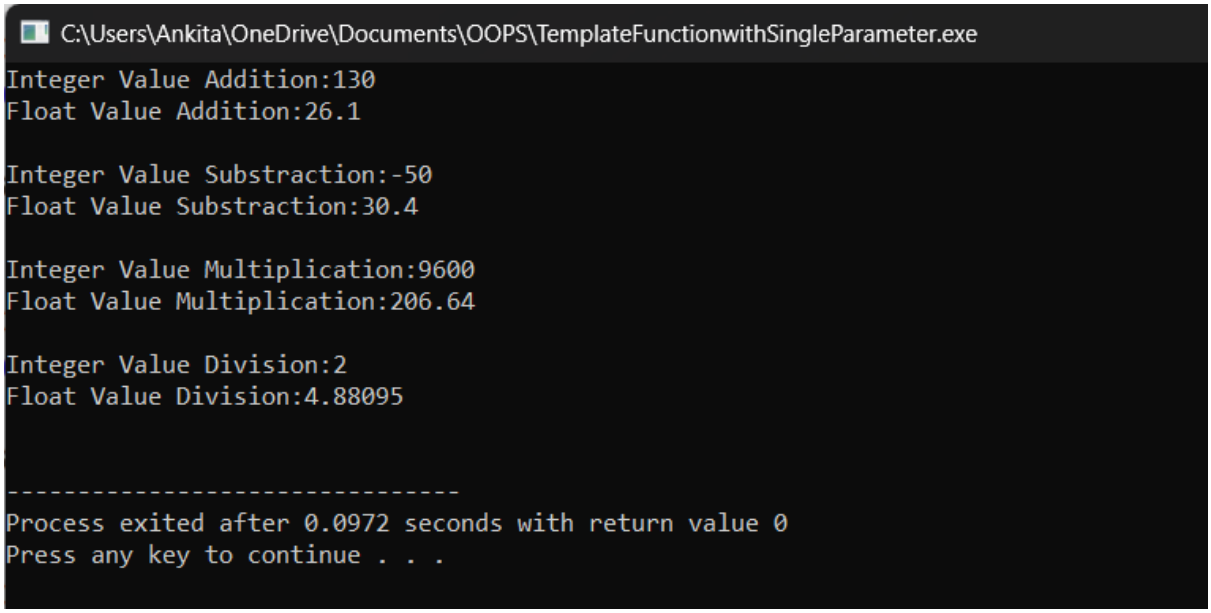
TEMPLATE FUNCTION WITH SINGLE PARAMETER:

INPUT:

```
#include<iostream>
using namespace std;
template<typename T>
T add(T num1,T num2, T num3)
{
   return(num1+num2+num3);
}

template<typename T1>
T1 sub(T1 num1,T1 num2, T1 num3)
{
   return(num1-num2-num3);
}

template<typename T2>
T2 mul(T2 num1,T2 num2, T2 num3)
{
   return(num1*num2*num3);
}

template<typename T3>
T3 div(T3 num1,T3 num2)
{
   return(num1/num2);
}

int main()
```

```
{
    int add1,sub1,mul1,div1;
    double add2,sub2,mul2,div2;

    add1=add<int>(40,80,10);
    cout<<"Integer Value Addition:"<<add1<<endl;

    add2=add<float>(20.5,4.2,1.4);
    cout<<"Float Value Addition:"<<add2<<endl<<endl;

    sub1=sub<int>(40,80,10);
    cout<<"Integer Value Substraction:"<<sub1<<endl;

    sub2=sub<float>(50.6,18.7,1.5);
    cout<<"Float Value Substraction:"<<sub2<<endl<<endl;

    mul1=mul<int>(40,80,3);
    cout<<"Integer Value Multiplication:"<<mul1<<endl;

    mul2=mul<float>(20.5,4.2,2.4);
    cout<<"Float Value Multiplication:"<<mul2<<endl<<endl;

    div1=div<int>(40,20);
    cout<<"Integer Value Division:"<<div1<<endl;

    div2=div<float>(20.5,4.2);
    cout<<"Float Value Division:"<<div2<<endl<<endl;
    return 0;
}
```

OUTPUT:



TEMPLATE FUNCTION WITH MULTIPLE PARAMETER:

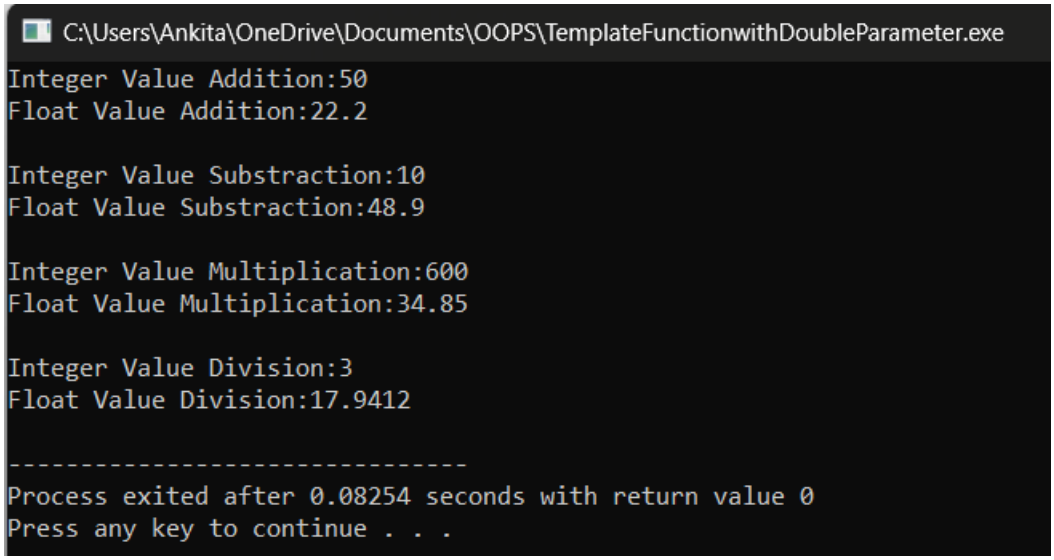
INPUT:

```
#include<iostream>
using namespace std;
template<typename T1,typename T2>
T1 add(T1 num1,T2 num2)
{
   return(num1+num2);
}

template<typename s1,typename s2>
s1 sub(s1 num1,s2 num2)
{
   return(num1-num2);
}
```

```
template<typename m1,typename m2>
m1 mul(m1 num1,m2 num2)
{
   return(num1*num2);
}

template<typename d1,typename d2>
d1 div(d1 num1,d2 num2)
{
   return(num1/num2);
}
int main()
{
   int add1,sub1,mul1,div1;
   double add2,sub2,mul2,div2;

   add1=add<int,int>(30,20);
   cout<<"Integer Value Addition:"<<add1<<endl;

   add2=add<float,float>(20.5,1.7);
   cout<<"Float Value Addition:"<<add2<<endl<<endl;

   sub1=sub<int,int>(30,20);
   cout<<"Integer Value Substraction:"<<sub1<<endl;

   sub2=sub<float,float>(50.6,1.7);
   cout<<"Float Value Substraction:"<<sub2<<endl<<endl;
```

```
    mul1=mul<int,int>(30,20);
    cout<<"Integer Value Multiplication:"<<mul1<<endl;

    mul2=mul<float,float>(20.5,1.7);
    cout<<"Float Value Multiplication:"<<mul2<<endl<<endl;

    div1=div<int,int>(90,30);
    cout<<"Integer Value Division:"<<div1<<endl;

    div2=div<float,float>(30.5,1.7);
    cout<<"Float Value Division:"<<div2<<endl;
    return 0;
}
```

OUTPUT:



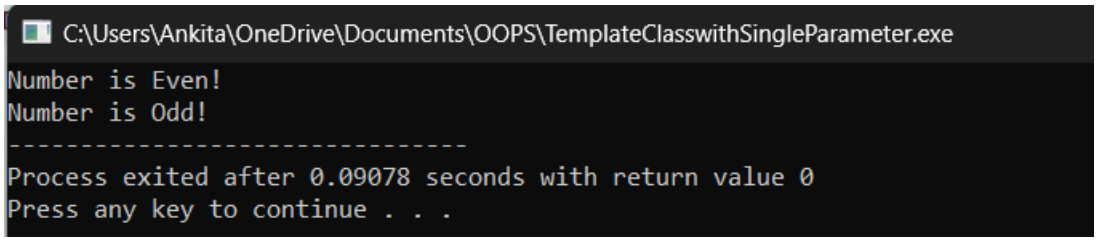
TEMPLATE CLASS WITH SINGLE PARAMETER

INPUT:

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

template <class T>
class demo{
	private:
		T num;
	public:
	demo(T n1)
	{
		num=n1;
	}

	void check()
	{
		if(num%2==0)
		{
			cout<<"Number is Even!";
		}
		else
		{
			cout<<"Number is Odd!";
		}
	}
};
int main()
{
```

```
	demo<int>obj1(6);
	obj1.check();
	cout<<endl;
	demo<int>obj2(19);
	obj2.check();
	return 0;
}
```

OUTPUT:



TEMPLATE CLASS WITH MULTIPLE PARAMETER:

INPUT:

```
#include<iostream>
using namespace std;
template<class T1, class T2>
class demo{
	private:
		T1 num1;
		T2 num2;
	public:
	demo(T1 length, T2 breadth)
	{
		num1=length;
```

```
            num2=breadth;
        }

        void area()
        {
            cout<<"Area of Rectangle:"<<(num1*num2)<<endl;
        }
};

int main()
{
        demo<int, int>obj1(9,4);
        obj1.area();
        demo<float, float>obj2(5.2,7.7);
        obj2.area();
        return 0;

}
```

OUTPUT: