Lab-11

Title: Template and Exception handling

Objective:

• To be familiar with class template and function template
• To be familiar with exception handling mechanism and implement them whenever necessary

Theory:

• Introduction to generic programming
• Function template with their syntax
• Class template with their syntax
• Exception handling mechanism in C++

1) Create a function template to find the sum of two integers and float using function template.

```
#include<iostream>
using namespace std;
template <class T1>
void Sum(T1 a,T1 b)
{
        cout<<"Sum="<<(a+b)<<endl;
};
int main(){
int i1=5,i2=6;
float f1=5.5,f2=6.6;
Sum(i1,i2);
Sum(f1,f2);
return 0;
}
```

2) Write a program to find the sum and product of two integer and float using function template.

```
#include<iostream>
using namespace std;
template <class T1>
void Sum(T1 a,T1 b)
{
     cout<<"Sum="<<(a+b)<<endl;
     cout<<"Product="<<(a*b)<<endl;
};
int main(){
     int i1=5,i2=6;
    float f1=5.5,f2=6.6;
    Sum(i1,i2);
    Sum(f1,f2);
    return 0;
}
```

3) Write a C++ program to find the largest among two integers, two characters and float using function template.

```
#include<iostream>
using namespace std;
template<class T1>
void compare(T1 a,T1 b){
        if(a>b)
            cout<<"Maximum value:"<<a<<endl;
        else
             cout<<"Maximum value:"<<b<<endl;
};
         int main(){
         int i1=4,i2=5;
         float f1=4.4,f2=5.5;
         char c1='a',c2='b';
         cout<<"for integer number:"<<endl;
         compare(i1,i2);
         cout<<"for floating number:"<<endl;
         compare(f1,f2);
         cout<<"for character:"<<endl;
         compare(c1,c2);
         return 0;
}
```

4) Create a function template to swap two integers, two float and two characters.

```
#include<iostream>
using namespace std;
template<class T>
void swapg(T a,T b)
{
         T temp;
         temp=a;
         a=b;
         b=temp;
};
int main(){
         int i1=4,i2=9;
         float f1=4.4,f2=9.9;
         cout<<"Before swapping value:"<<endl;
         cout<<"i1:"<<i1<<" i2:"<<i2<<endl;
         cout<<"f1:"<<f1<<" f2:"<<f2<<endl;
         swapg(i1,i2);
         swapg(f1,f2);
         cout<<"After swapping value:"<<endl;
```

```
        cout<<"i1:"<<i1<<" i2:"<<i2<<endl;
        cout<<"f1:"<<f1<<" f2:"<<f2<<endl;
        return 0;
}
```

5) Write a program to find the roots of the quadratic equation using function template.

```
#include<iostream>
#include<math.h>
using namespace std;
template<class T>
void calculate(T a,T b,T c)
{
        T d=b*b-4*a*c;
        if(d<0){
                cout<<"Root are imaginary"<<endl;
        }
        else if(d==0){
                cout<<"Root are real and equal"<<endl;
                cout<<"R1=R2:"<<(-b/(2.0*a));
        }
        else{
                cout<<"Root are reasl and unequal"<<endl;
                float r1=(-b+sqrt(d))/(2.0*a);
                float r2=(-b-sqrt(d))/(2.0*a);
                cout<<"R1="<<r1<<endl;
                cout<<"R2="<<r2<<endl;
}
};
int main(){
        int a1,b1,c1;
        float a2,b2,c2;
        cout<<"Enter the integer coefficient "<<endl;
        cin>>a1>>b1>>c1;
        calculate(a1,b1,c1);
        cout<<"Enter the integer coefficient "<<endl;
        cin>>a2>>b2>>c2;
        calculate(a2,b2,c2);
        return 0;
}
```

6) Create a class template to find the sum and average of two integers and two float using class template.

```
#include<iostream>
using namespace std;
template <class T1>
class Sum{
private:
        T1 a;
        T1 b;
public:
Sum(T1 x,T1 y){
        a=x;
        b=y;
}
void add(){
        cout<<"Sum="<<(a+b)<<endl;
        cout<<"Average="<<(a+b)/2.0<<endl;
}
};
int main(){
        Sum<int>sum1(5,6);
        Sum<float>sum2(5.5,6.6);
        sum1.add();
        sum2.add();
        return 0;
}
```

7) Write a program to illustrate exception handling mechanism in C++.

```
#include<iostream>
using namespace std;
int main(){
        int a,b,x;
        cout<<"Enter values of and b"<<endl;
        cin>>a>>b;
        x=a-b;
        try
        {
                if(x!=0)
                {
                cout<<"Result(a/x):"<<a/x<<endl;
                }
                else
                {
                throw(x);
                }
        }
```

```
	catch(int i)
	{
		cout<<"Exception caught:DIVIDE BY ZERO"<<endl;
	}
	cout<<"END";
	return 0;
}
```
8) Write a program that catches multiple exceptions. (Handling Multiple exceptions)
```
#include<iostream>
using namespace std;
void test(int x)
{
try
	{
	if(x==1){
		throw x;
	}
	else if(x==0){
		throw 'x';
	}
	else if(x==-1){
		throw 1.0;
	}
	cout<<"End of try-block"<<endl;
	}
catch(char c)
{
	cout<<"Caught a character"<<endl;
}
catch(int m)
{
	cout<<"Caught a integer"<<endl;
}
catch(double d)
{
	cout<<"Caught a double"<<endl;
}
cout<<"End of try-catch system"<<endl;
}
int main(){
	cout<<"Testing multiple catches"<<endl;
	cout<<"x==1"<<endl;
	test(1);
	cout<<"x==0"<<endl;
```

```
        test(0);
        cout<<"x==-1"<<endl;
        test(-1);
        cout<<"x==2"<<endl;
        test(2);
        return 0;
}
```