Practical-1

Aim: **WAP to create a class which illustrate the concept of handling all types of exceptions using general exception.**

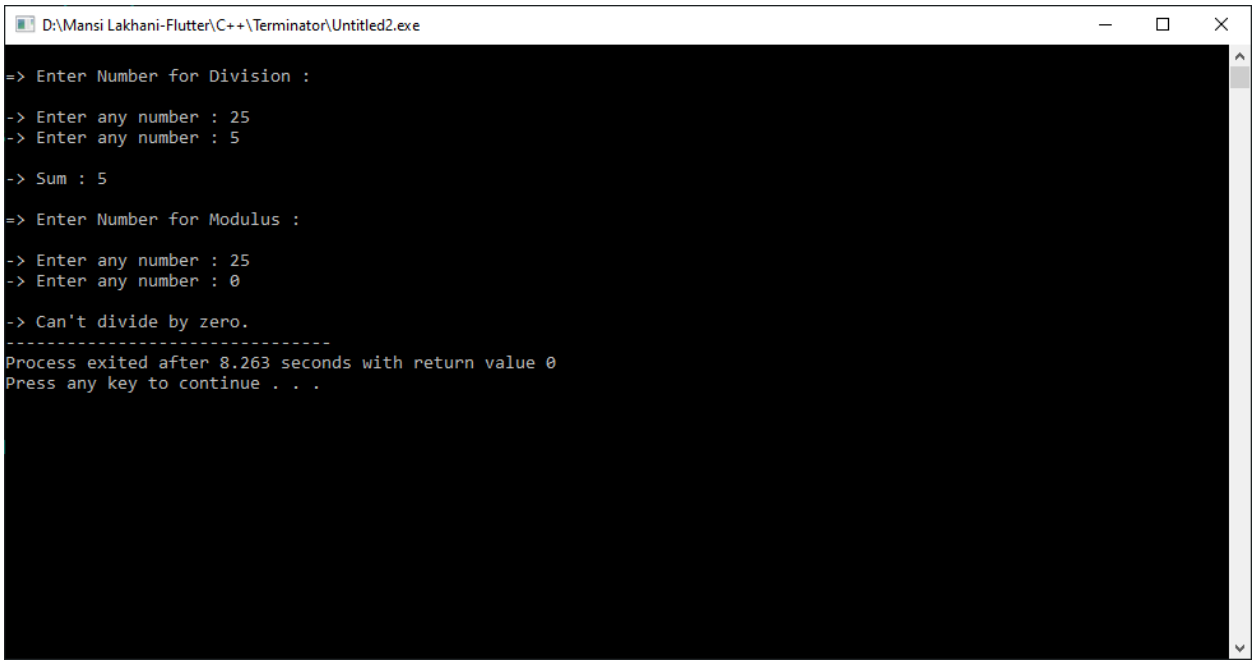
Program:

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

class Division
{
	public :
		int x;

	public :
		void setData()
		{
			cout << "-> Enter any number : ";
			cin >> this->x;
		}
		void getData()
		{
			cout <<endl<<"-> Sum : "<<this->x;
		}
		Division operator/(Division n)
		{
			Division  temp;
			temp.x = this->x / n.x;
			return temp;
		}

};
class Modulus
{
	public :
		int y;
```

```
	public :
		void setData()
		{
			cout << "-> Enter any number : ";
			cin >> this->y;
		}
		void getData()
		{
			cout <<endl<<"-> Sum : "<<this->y <<endl;
		}
		Modulus  operator%(Modulus  n)
		{
			Modulus   temp;
			temp.y = this->y % n.y;
			return temp;
		}

};

int main()
{
	Division d1 ,d2,d3;
	Modulus m1,m2,m3;

	cout <<endl <<"=> Enter Number for Division : " <<endl <<endl ;
	d1.setData();
	d2.setData();

	try
	{
		if(d2.x==0)
		{
			throw 26;
		}
		else
		{
			d3 = d1/d2;
			d3.getData();

		}
```

```
	}
	catch(...)
	{
		cout <<endl <<"-> Can't divide by zero.";
	}

	cout <<endl <<endl <<"=> Enter Number for Modulus : " <<endl <<endl ;

	m1.setData();
	m2.setData();

	try
	{
		if(m2.y==0)
		{
			throw 5;
		}
		else
		{
			m3 = m1%m2;
			m3.getData();

		}
	}
	catch(...)
	{
		cout <<endl <<"-> Can't divide by zero.";
	}

	return 0;
}
```

Output:


D:\Mansi Lakhani-Flutter\C++\Terminator\Untitled2.exe
=> Enter Number for Division :
-> Enter any number : 25
-> Enter any number : 5
-> Sum : 5
=> Enter Number for Modulus :
-> Enter any number : 25
-> Enter any number : 0
-> Can't divide by zero.

Process exited after 8.263 seconds with return value 0
Press any key to continue . . .