EXP.NO: 4.1

AIM: Write a c++ program to illustrate the use of function templates

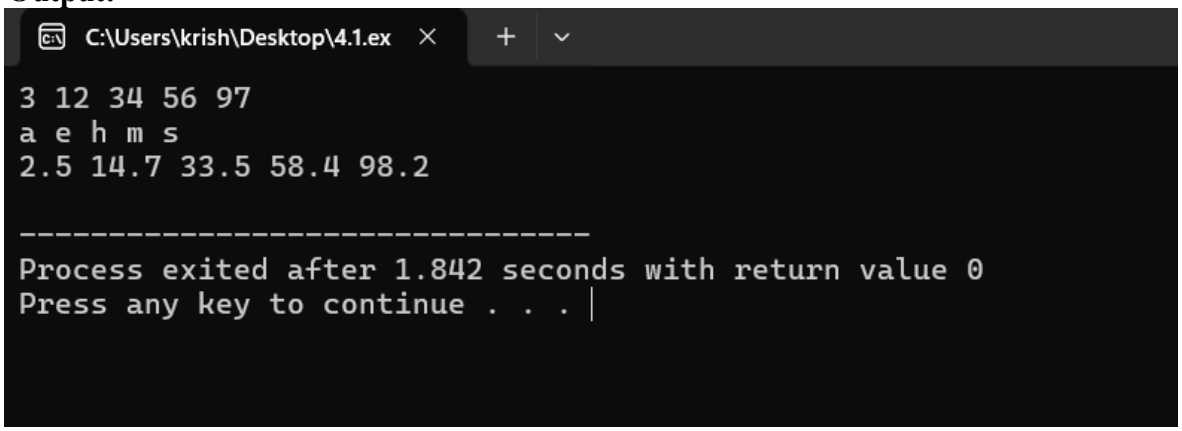
PROGRAM:

```
#include<iostream>
using namespace std;
template <typename T> //generic type T
void sort(T a[],int n)
{
	T temp;
	int i,j;
	for(i=0;i<n-1;i++)
	{
		for(j=0;j<n-1-i;j++)
		{
			if(a[j]>a[j+1])
			{
				temp=a[j]; //swapping
				a[j]=a[j+1];
				a[j+1]=temp;
			}
		}
	}
}
template<typename T>
void print(T a[],int n)
{
	int i,j;
	for(i=0;i<n;i++)
	{
```

```
            cout<<a[i]<<" ";


        }
        cout<<endl;
}
int main()
{
        int a[5]={12,97,34,56,3};
        char c[5]={'s','e','a','m','h'};
        float f[5]={2.5,14.7,98.2,33.5,58.4};
        sort(a,5);
        print(a,5);
        sort(c,5);
        print(c,5);
        sort(f,5);
        print(f,5);
        return 0;
}
```

Output:



EXP.NO: 4.2

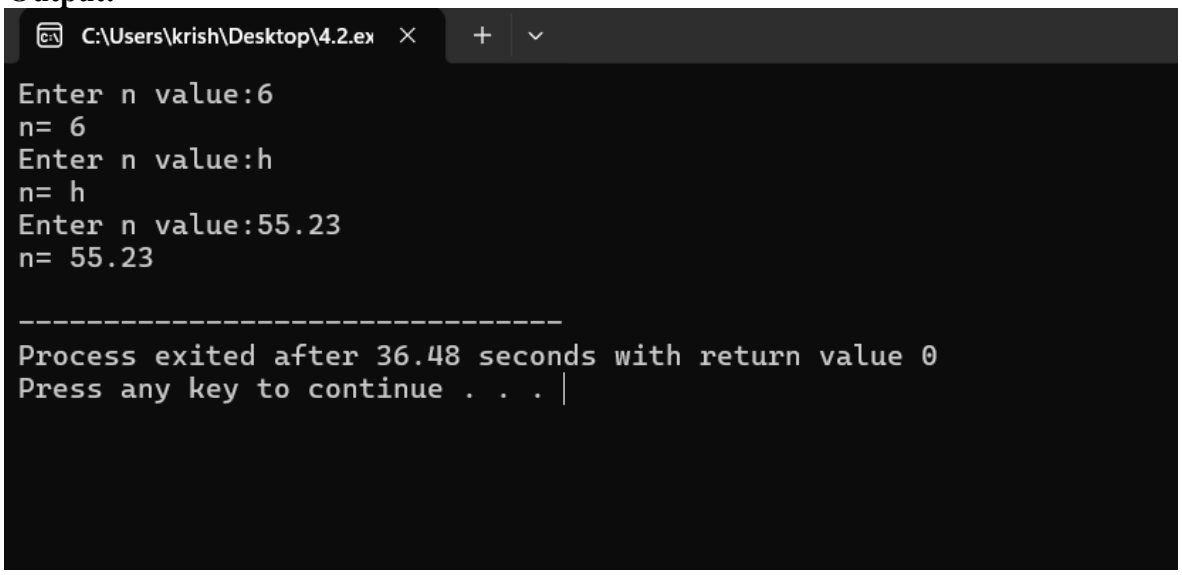
AIM: : Write a c++ program to implement template class

PROGRAM:

```
#include<iostream>
using namespace std;
template <class T> // template class definition
class Sample
{
	private:
		T n; // variable of generic type T
	public:
		void get()
		{
			cout << "Enter n value:";
			cin >> n;
		}
		void show()
		{
			cout << "n= " << n << endl;
		}
};
int main()
{
	Sample<int> s1;
	s1.get(); //call of get function
	s1.show(); //call of show function
	Sample<char> s2;
	s2.get();  //call of get function
	s2.show(); //call of show function
```

```
    Sample<float> s3;
    s3.get(); //call of get function
    s3.show(); //call of show function
    return 0; // end of program
}
```

Output:



EXP.NO: 4.3

AIM: Write a c++ program to implement class templates with multiple parameters

PROGRAM:

```
#include<iostream>
using namespace std;
template <class T1, class T2> // template class definition  with multiple generic types

class Sample
{
	private:
		T1 x; //variable of T1 type
		T2 y; //variable of T2 type
	public:
		void get()
		{
			cin >> x >> y;
		}
		void show()
		{
			cout << "x= " << x << endl;
			cout << "y= " << y << endl;
		}
};
int main()
{
	Sample<int, float> s1; //object of int and float
	cout << "Enter int , float value:";
	s1.get();
	s1.show();
```

```
        Sample<char, int> s2;  //object of char and int
        cout << "Enter char , int value:";
        s2.get();
        s2.show();
        Sample<float, char> s3;  //object of float and char
        cout << "Enter float, char value:";
        s3.get();
        s3.show();
        return 0;
}
```

Output: