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

int square(int x)
{
	return x*x;
}


int main()
{
	int x=2;
	cout<<square(x);
}
```

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

int square(int x)
{
	return x*x;
}

float square(float x)
{
	return x*x;
}

int main()
{
	int x=2;
	cout<<square(x)<<endl;

	float y=2.2f;
	cout<<square(y);
}
```

//Function overloading:

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

int square(int x)
{
	return x*x;
}

float square(float x)
{
	return x*x;
}
double square(double x)
{
	return x*x;
}

int main()
{
	int x=2;
	cout<<square(x)<<endl;

	float y=2.2f;
	cout<<square(y)<<endl;

	double z=2.5;
	cout<<square(z);
}
```

Generic programming: concept

Templates: tool

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

template<class T>
T square (T x)
{
```

```
	T result;
	result=x*x;
	return result;
}

int main()
{
	int i=2;
	int ii=square(i);    //implicit template expression

	float f=2.5f;
	float ff=square(f);  //implicit template expression

	double d=3.5;
	double dd=square(d);

	cout<<ii<<endl<<ff<<endl<<dd;
}
```

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

template<class T>
T square (T x)
{
	T result;
	result=x*x;
	return result;
}

int main()
{
	int i=2;
	int ii=square<int>(i);      //explicit template expression

	float f=2.5f;
	float ff=square<float>(f); //explicit template expression

	double d=3.5;
	double dd=square(d);       //implicit template expression
```

```
	char c='A';
	char cc=square(c);

	cout<<ii<<endl<<ff<<endl<<dd<<endl<<cc;
}
```

//Multiple parameters in function template.. all parameters are of same type

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

template<class T>
T sum(T x, T y)
{
	return (x+y);
}

int main()
{
	cout<<sum(2,4)<<endl;
	cout<<sum(2.4,5.66)<<endl;
	cout<<sum(5.6f,7.8f);
}
```

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

template<class T>
T sum(T x, T y, T z)
{
	return (x+y+z);
}

int main()
{
	cout<<sum(2,4,7)<<endl;		//implicit template expression
	cout<<sum(2.4,5.66,7.4)<<endl;
	cout<<sum(5.6f,7.8f,3.7f);
}
```

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

template<class T>
T sum(T x, T y, T z)
{
	return (x+y+z);
}

int main()
{
	cout<<sum<int>(2,4,7)<<endl;		//explicit template expression
	cout<<sum<double>(2.4,5.66,7.4)<<endl;
	cout<<sum<float>(5.6f,7.8f,3.7f);
}
```

//Function template accepting multiple types of parameters..

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

template<class T, class U>
void sum(T x, U y)
{
	cout<<(x+y)<<endl;
}

int main()
{
	sum<int,int>(2,4);
	sum<double,double>(2.4,5.66);
	sum<float,float>(5.6f,7.8f);

	sum(2,5.6);
	sum(2.4f,5);
	sum(6,4.6f);
}
```

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

template<class T, class U>
T sum(T x, U y)
{
	return (x+y);
}

int main()
{
	cout<<sum<int,int>(2,4)<<endl;
	cout<<sum<double,double>(2.4,5.66)<<endl;
	cout<<sum<float,float>(5.6f,7.8f)<<endl;

	cout<<sum<int,double>(2,5.6)<<endl;    //7
	cout<<sum<float,int>(2.4f,5)<<endl;
	cout<<sum<int,float>(6,4.6f)<<endl;     //10
}
```

O/p:
6
8.06
13.4
7
7.4
10

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

template<class T, class U, class V>
V sum(T x, U y)
{
	return (x+y);
}

int main()
{
```

```
	cout<<sum<int,double,double>(2,5.6);
	cout<<sum<float,int,float>(2.4f,5);
	cout<<sum<int,float,float>(6,4.6f);
}
```

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

template<class T>
void swap_val(T &x, T &y) //pass the reference.. for the modifications to be reflected in the array
{
	T temp;
	temp=x;
	x=y;
	y=temp;
}

template<class X>
void sort(X arr[], int n)
{
	for(int i=0;i<n-1;i++)

	  for(int j=0;j<n-i-1;j++)
	       if(arr[j]>arr[j+1])
	        swap_val(arr[j],arr[j+1]);
}

template<class P>
void print(P arr[], int n)
{
	for(int i=0;i<n;i++)
	cout<<arr[i]<<"  ";

	cout<<endl;
}

int main()
{
	int a[5]={4,7,2,1,3};
	sort(a,5);
	print(a,5);
```

```
	float f[6]={1.5,6.7,3.3,8.9,2.5,10.5};
	sort(f,6);
	print(f,6);

	char c[4]={'a','h','f','b'};
	sort(c,4);
	print(c,4);
}
```

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

template<class T>
void swap_val(T &x, T &y)
{
	T temp;
	temp=x;
	x=y;
	y=temp;
}

template<class X>
void sort(X arr[], int n)
{
	for(int i=0;i<n-1;i++)

	 for(int j=0;j<n-i-1;j++)
	     if(arr[j]>arr[j+1])
	      swap_val<X>(arr[j],arr[j+1]); //from a function temnplate, call another function
template
}

template<class P>
void print(P arr[], int n)
{
	for(int i=0;i<n;i++)
	cout<<arr[i]<<"  ";

	cout<<endl;
}
```

```
int main()
{
	int a[5]={4,7,2,1,3};
	sort<int>(a,5);
	print<int>(a,5);

	float f[6]={1.5,6.7,3.3,8.9,2.5,10.5};
	sort<float>(f,6);
	print<float>(f,6);

	char c[4]={'a','h','f','b'};
	sort<char>(c,4);
	print(c,4);
}
```

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

template<class T>
T max_arr(T x[], int n)
{
	T max=x[0];
	for(int i=1;i<n;i++)
	if(x[i]>max)
	{
		max=x[i];
	}
	return max;
}

int main()
{
	int a[]={5,2,8,7};
	cout<<max_arr(a,4)<<"  ";

	float f[]={7.8,2.4,1.7,9.7,3.4};
	cout<<max_arr(f,5)<<"   ";

	char c[]={'r','q','b'};
	cout<<max_arr(c,3);
```

```
}
```

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

template<int n,class T>
void loop(T x)
{
	for(int i=1;i<=n;i++)
	cout<<x<<"  ";
}

int main()
{
	loop<10,int>(5);

	loop<5,double>(4.5);
}
```

```
//A function template accepting all non-type parameters..

#include<iostream>
using namespace std;

template<int n, int x>
void loop(int y)
{
	for(int i=1;i<=n;i++)
	cout<<y<<"  "<<x;
}

int main()
{
	//loop<10,int>(5);

	//loop<5,double>(4.5);

	loop<5,6>(7);
```

```
}
```

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

template<class T, int n>
void mult(T x)
{
	for(int i=1;i<=n;i++)
	{
		T res=x*n;
		cout<<res<<"  ";
	}
	cout<<"\n";
}

int main()
{
	int x=2;

	mult<int,5>(x);
	mult<double,3>(4.5);
	mult<float,4>(1.5f);

}
```

```
//Default values in function template...

#include<iostream>
using namespace std;

template<class T, int n=5>
void mult(T x)
{
	for(int i=1;i<=n;i++)
	{
		T res=x*n;
		cout<<res<<"  ";
	}
	cout<<"\n";
```

```
}

int main()
{
	int x=2;

	mult<int>(x);
	mult<double,3>(4.5);
	mult<float>(1.5f);

}
```

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

template<class T=int, int n=5>
void mult(T x)
{
	for(int i=1;i<=n;i++)
	{
		T res=x*n;
		cout<<res<<"  ";
	}
	cout<<"\n";
}

int main()
{
	int x=2;

	mult<>(x);
	mult<double>(4.5);
	mult<float,5>(1.5f);

}
```

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

```
template<int n=5, class T=int>
void mult(T x)
{
	for(int i=1;i<=n;i++)
	{
		T res=x*n;
		cout<<res<<"  ";
	}
	cout<<"\n";
}

int main()
{
	int x=2;

	mult<>(x);
	mult<3,double>(4.5);
	mult<5,float>(1.5f);

}
```

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

template<class T, int n>
void loop(T x)
{
   for(int i=1;i<=n;i++)
    cout<<x<<endl;
}

int main()
{
   //loop(5);

   //loop<int>(6);

   //loop<7>(6.5f);

   loop<double,3>(5.55);
}
```

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

template<class T, int n=10>
void loop(T x)
{
   for(int i=1;i<=n;i++)
    cout<<x<<endl;
}

int main()
{
   //loop(5);   //OK

   //loop<int>(6); //OK

   //loop<7>(6.5f); //wrong

   loop<double,3>(5.55); //OK
}
```

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

template<int n=10, class T=int>
void loop(T x)
{
   for(int i=1;i<=n;i++)
    cout<<x<<endl;
}

int main()
{
   //loop(5);   //OK

  // loop<int>(6); //wrong

   //loop<7>(6.5f); //OK
```

```
    //loop<double,3>(5.55); //wrong
}
```

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

template<int n, class T=int>
void loop(T x)
{
    for(int i=1;i<=n;i++)
     cout<<x<<endl;
}

int main()
{
    //loop(5);      //wrong

    //loop<int>(6); //wrong

    //loop<7>(6.5f); //OK

    //loop<double,3>(5.55); //wrong
}
```

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

template<int n=5, class T>
void loop(T x)
{
    for(int i=1;i<=n;i++)
     cout<<x<<endl;
}

int main()
{
    //loop(5);   //OK
```

```
    //loop<int>(6); //wrong

    //loop<7>(6.5f); //OK

    //loop<double,3>(5.55); //wrong
}
```

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

template<class U=int, class T>
void sum(T x, U y)
{
    cout<<x+y;
}

int main()
{
    sum(2,4);

}
```

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

template<class U=int, class T>
void sum(T x, U y)
{
    cout<<x+y;
}

int main()
{
    sum<int>(2,3);

}
```

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

template<class U=int, class T>
void sum(U  x, T y)
{
   cout<<x+y;
}

int main()
{
   sum<int,double>(2,3.5);

}
```

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

template<class U=int, class T>
void sum(U  x, T y)
{
   cout<<x+y;
}

int main()
{
   sum<double>(2,3);

}
```

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

template<class U=int, class T, class V>
V sum(U  x, T y)
{
   return x+y;
}

int main()
```

```
{
    cout<<sum<double, double,double>(2.5,3);

}
```
