Institute of Computer Technology
B. Tech Computer Science and Engineering
Subject: ESFP-II (2CSE203)

PRACTICAL-12

AIM: - To learn about creation of function template and class template in C++.

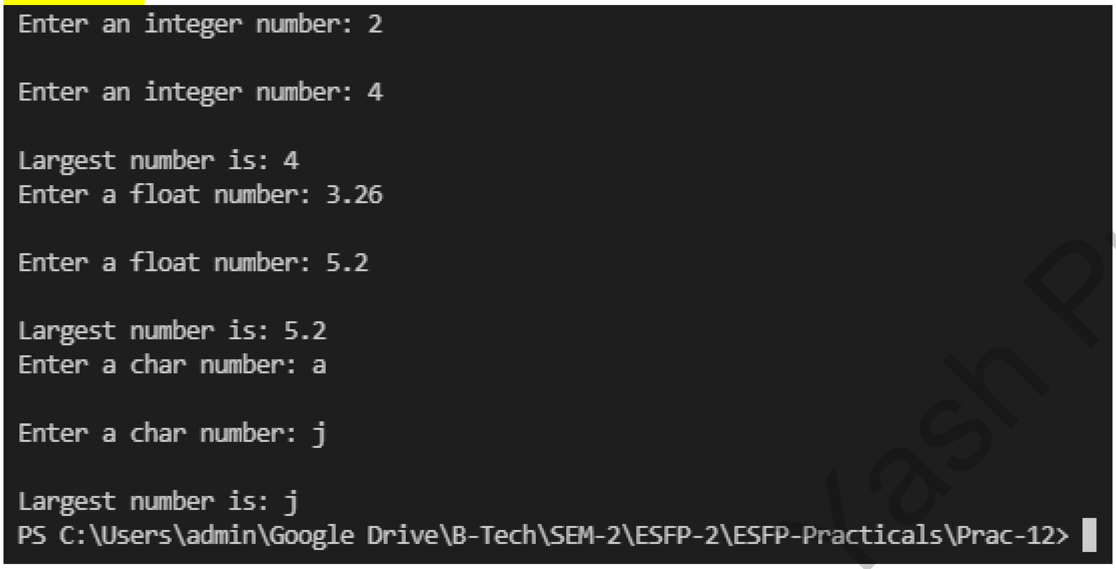
1. Simran wants to implement a program in C++ to create a template which finds the largest among two integer, floating values & character values.

CODE:

```
#include <iostream>
using namespace std;
template <typename Y>
void Large(Y a, Y b)
{
   Y largest;
   largest=0;
   if (a>b)
   {
      largest=a;
   }
   else
   {
      largest=b;
   }
   cout<<"\nLargest number is: "<<largest;
}
int main()
{

   int a,b;
   float c,d;
   char e,f;
   cout<<"\nEnter an integer number: ";
   cin>>a;
   cout<<"\nEnter an integer number: ";
   cin>>b;
   Large(a,b);
   cout<<"\nEnter a float number: ";
   cin>>c;
   cout<<"\nEnter a float number: ";
   cin>>d;
   Large(c,d);
   cout<<"\nEnter a char number: ";
```

```
    cin>>e;
    cout<<"\nEnter a char number: ";
    cin>>f;
    Large(e,f);
    return 0;
}
```

OUTPUT:



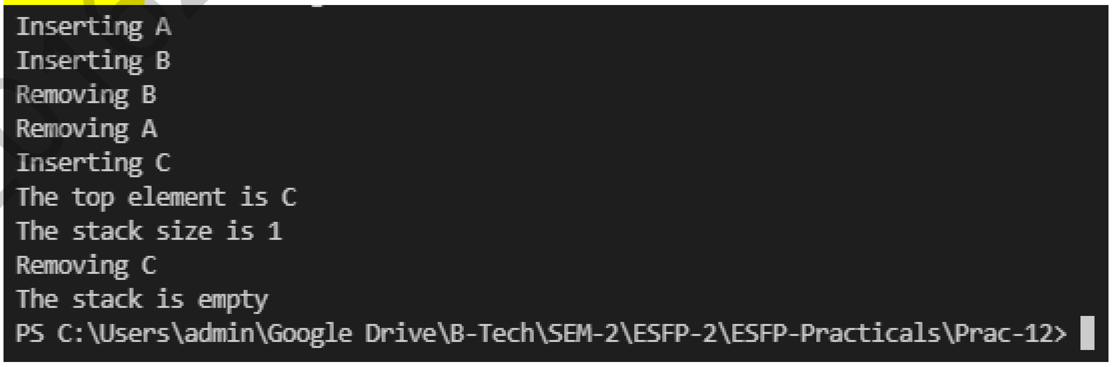
```
Enter an integer number: 2

Enter an integer number: 4

Largest number is: 4
Enter a float number: 3.26

Enter a float number: 5.2

Largest number is: 5.2
Enter a char number: a

Enter a char number: j

Largest number is: j
PS C:\Users\admin\Google Drive\B-Tech\SEM-2\ESFP-2\ESFP-Practicals\Prac-12>
```

2. Create one class template and use it for to implement PUSH & POP operations of stack.

CODE:

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

#define SIZE 10
template <class X>
class stack
{
    X *arr;
    int top;
    int capacity;

public:
    stack(int size = SIZE);
    void push(X);
    X pop();
    X peek();

    int size();
    bool isEmpty();
    bool isFull();
    ~stack() {
        delete[] arr;
    }
};
template <class X>
stack<X>::stack(int size)
{
    arr = new X[size];
    capacity = size;
    top = -1;
}
template <class X>
void stack<X>::push(X x)
{
    if (isFull())
    {
        cout << "Overflow\nProgram Terminated\n";
        exit(EXIT_FAILURE);
    }

    cout << "Inserting " << x << endl;
    arr[++top] = x;
}
template <class X>
X stack<X>::pop()
{
    if (isEmpty())
    {
        cout << "Underflow\nProgram Terminated\n";
        exit(EXIT_FAILURE);
    }

    cout << "Removing " << peek() << endl;
return arr[top--];
}
template <class X>
X stack<X>::peek()
{
    if (!isEmpty()) {
        return arr[top];
    }
    else {
        exit(EXIT_FAILURE);
    }
}
template <class X>
int stack<X>::size() {
    return top + 1;
}
template <class X>
bool stack<X>::isEmpty() {
    return top == -1;             // or return size() == 0;
}
template <class X>
bool stack<X>::isFull() {
    return top == capacity - 1;     // or return size() == capacity;
}
int main()
{
    stack<string> pt(2);
     pt.push("A");
    pt.push("B");
     pt.pop();
    pt.pop();
     pt.push("C");
  cout << "The top element is " << pt.peek() << endl;
    cout << "The stack size is " << pt.size() << endl;
    pt.pop();
    if (pt.isEmpty()) {
        cout << "The stack is empty\n";
    }
    else {
        cout << "The stack is not empty\n";
    }
    return 0;
}
```

OUTPUT:



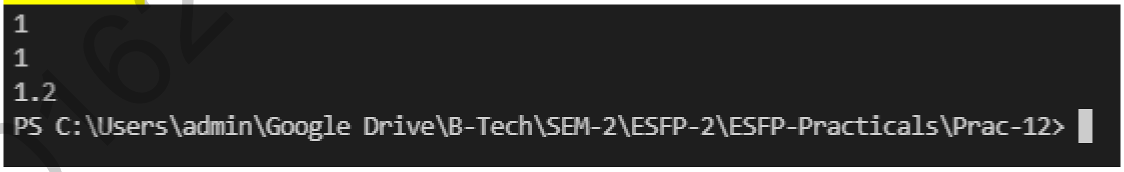
```
Inserting A
Inserting B
Removing B
Removing A
Inserting C
The top element is C
The stack size is 1
Removing C
The stack is empty
PS C:\Users\admin\Google Drive\B-Tech\SEM-2\ESFP-2\ESFP-Practicals\Prac-12>
```

3. Demonstrate a C++ Program to show Example of Static member variable of template class.

CODE:

```
#include <iostream>
using namespace std;
template<class T> class XYZ{
   public:
      void putPri();
      static T ipub;
   private:
      static T ipri;
};

template <class T>
void XYZ<T>::putPri()
{
   cout<< ipri++ <<endl;
}

template <class T> T XYZ<T>::ipub=1;
template <class T> T XYZ<T>::ipri=1.2;

int main()
{
   XYZ<int> aaa;
   XYZ<float> bbb;

   aaa.putPri();
   cout<<aaa.ipub<<endl;
   bbb.putPri();
   return 0;
}
```

OUTPUT:



```
1
1
1.2
PS C:\Users\admin\Google Drive\B-Tech\SEM-2\ESFP-2\ESFP-Practicals\Prac-12>
```

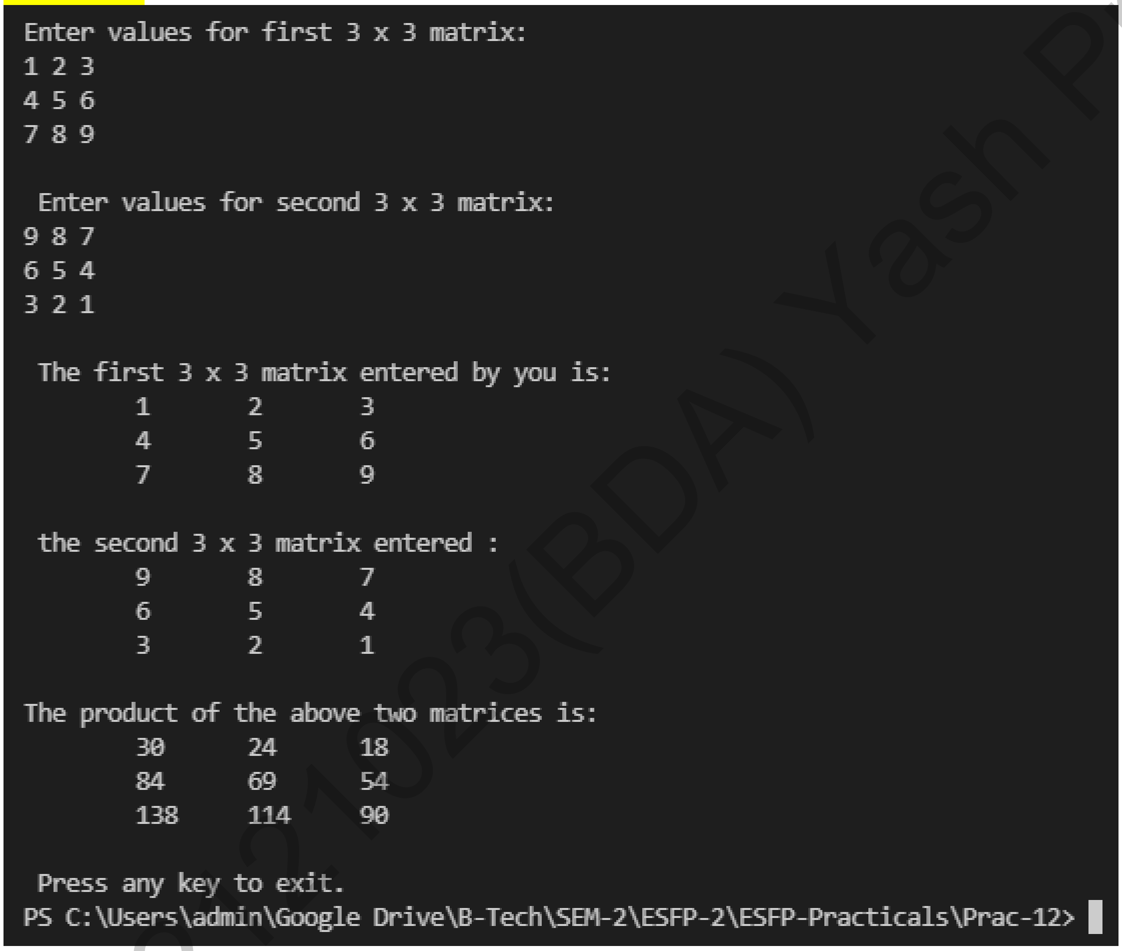
Post Practical Task

1. Implement a program in C++ template class to multiply 3 by 3 matrix with different data types.

CODE:

```
#include<iostream>
#include<conio.h>
using namespace std;
int main( )
{
int mat1 [3][3], mat2[3][3],mat3[3][3], i ,j, k, sum;
cout<<"\nEnter values for first 3 x 3 matrix:\n";
for ( i = 0 ; i <= 2 ; i++ )
{
for (j = 0 ; j <= 2 ; j++ )
cin>>mat1 [i][j] ;
}
cout<<"\n Enter values for second 3 x 3 matrix:\n";
for ( i = 0 ; i <= 2 ; i++ ){
for ( j = 0 ; j <= 2 ; j++ )
cin>>mat2[i][j] ;
}
cout<<"\n The first 3 x 3 matrix entered by you is:\n";
for ( i = 0 ; i <= 2 ; i++ )
{
for ( j = 0 ; j <= 2 ; j++ )
cout<<"\t"<< mat1[i][j] ;
cout<<"\n";
}
cout<<"\n the second 3 x 3 matrix entered :\n";
for ( i = 0 ; i <= 2 ; i++ )
{
for ( j = 0 ; j <= 2 ; j++ )
cout<<"\t"<< mat2[i][j] ;
cout<<"\n";
}
for ( i = 0 ; i <= 2 ; i++ )
{
for ( j = 0 ; j <= 2 ; j++ )
{
sum = 0;
for ( k = 0 ; k <=2 ; k++ )
sum = sum + mat1 [i][k] * mat2[k][j];
mat3[i][j] = sum ;
}
}
cout<<"\nThe product of the above two matrices is:\n";for ( i = 0 ;i<= 2 ; i++ )
{
for ( j = 0 ; j <= 2 ; j++ )
cout<<"\t"<<mat3[i][j] ;
cout<<"\n";
}
cout<<"\n Press any key to exit.";
return 0;
}
```

OUTPUT:



```
Enter values for first 3 x 3 matrix:
1 2 3
4 5 6
7 8 9

 Enter values for second 3 x 3 matrix:
9 8 7
6 5 4
3 2 1

 The first 3 x 3 matrix entered by you is:
        1       2       3
        4       5       6
        7       8       9

 the second 3 x 3 matrix entered :
        9       8       7
        6       5       4
        3       2       1

The product of the above two matrices is:
        30      24      18
        84      69      54
        138     114     90

 Press any key to exit.
PS C:\Users\admin\Google Drive\B-Tech\SEM-2\ESFP-2\ESFP-Practicals\Prac-12>
```

2. Predict the output?

```
#include <iostream>
using namespace std;
template <typename T>
void fun(const T&x)
{
static int count = 0;
cout << "x = " << x << " count = " << count << endl;
++count;
return;
}
int main()
{
fun<int> (1);
cout << endl;
fun<int>(1);
cout << endl;
fun<double>(1.1);
cout << endl;
return 0;
}
```

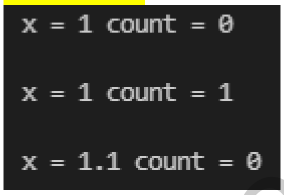
A) x = 1 count = 0
x = 1 count = 1
x = 1.1 count = 0

B) x = 1 count = 0
x = 1 count = 0
x = 1.1 count = 0

C) x = 1 count = 0
x = 1 count = 1
x = 1.1 count = 2

D) Compiler Error

OUTPUT:



```
x = 1 count = 0

x = 1 count = 1

x = 1.1 count = 0
```

3. Output of following program?
Assume that the size of int is 4 bytes and size of double is 8 bytes, and there is no alignment done by the compiler.

```
#include<iostream>
#include<stdlib.h>
using namespace std;
template<class T, class U, class V=double>
class A {
T x;
U y;
V z;
static int count;
};
int main()
```

```
{
A<int, int> a;
A<double, double> b;
cout << sizeof(a) << endl;
cout << sizeof(b) << endl;
return 0;
}
```

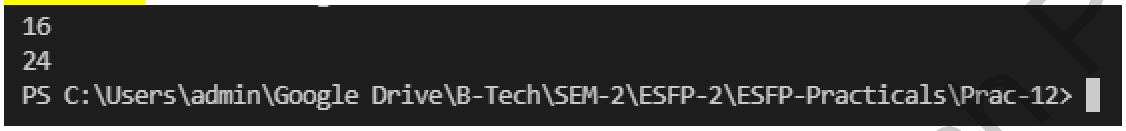
A. 16 24

B. 8 16

C. 20 28

D. Compiler Error: template parameters cannot have default values.

OUTPUT:



```
16
24
PS C:\Users\admin\Google Drive\B-Tech\SEM-2\ESFP-2\ESFP-Practicals\Prac-12>
```