Friend function

```cpp
#include <iostream>

class B;

class A
{
  public:
  void showB(B&);
};

class B
{
  private:
  int b;

  public:
  B() { b = 0; }
  friend void A::showB(B& x); // Friend function
};

void A::showB(B& x)
{
  // Since showB() is friend of B, it can
  // access private members of B
  std::cout << "B::b = " << x.b;
}

int main()
{
  A a;
  B x;
  a.showB(x);
  return 0;
}
```
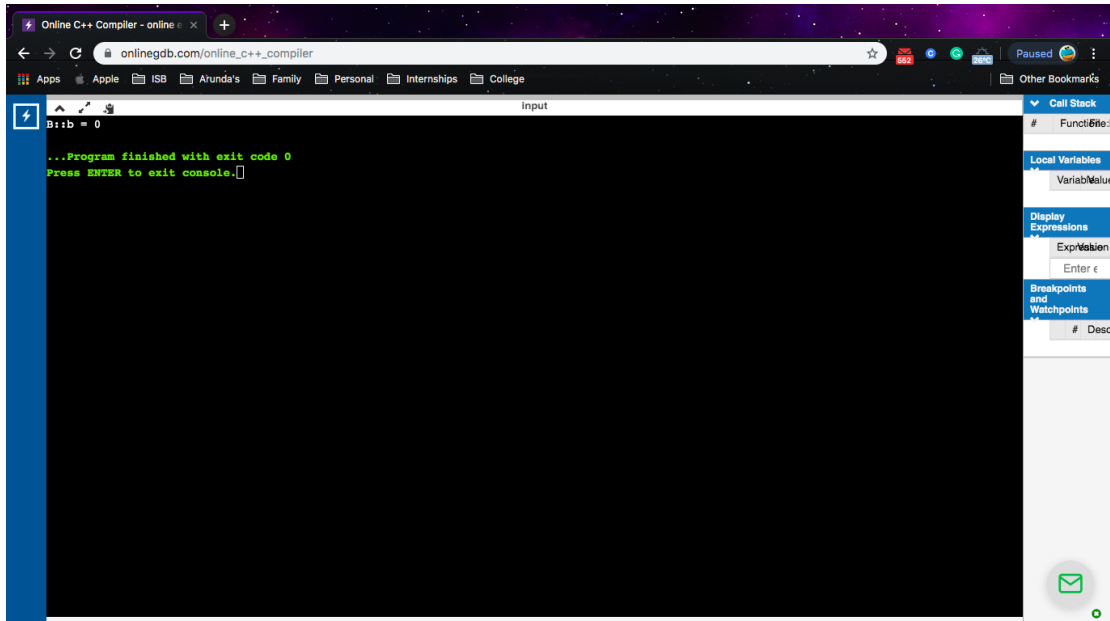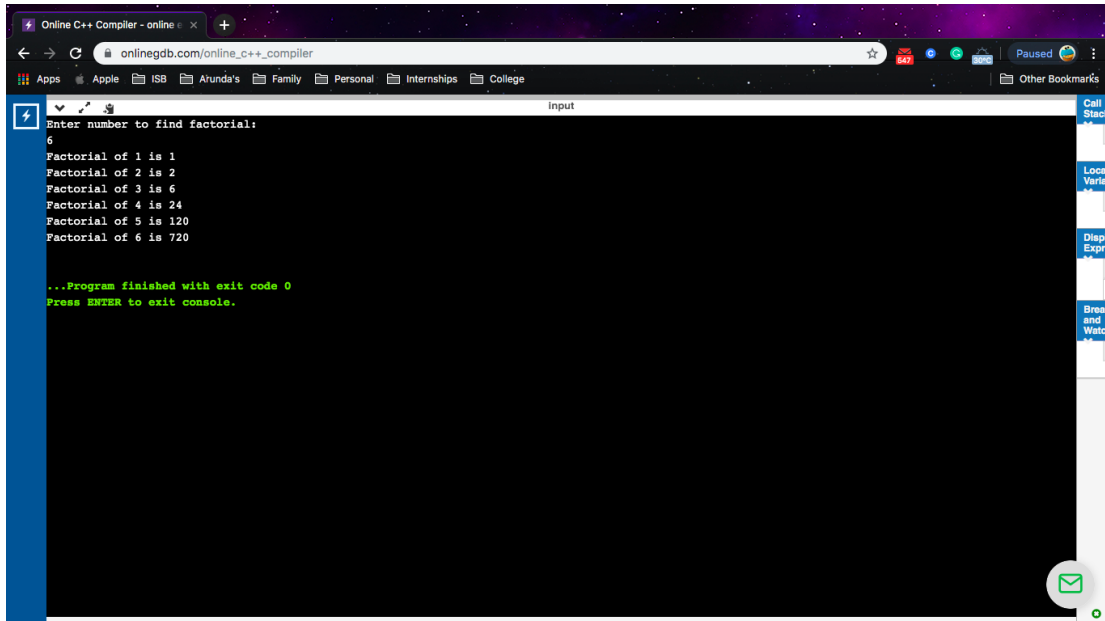
Output

Inline function

```cpp
#include <iostream>
#include <conio.h>
using namespace std;
inline void factorial(int x)
{
    int i,f=1,p=1;
    for(i=1;i<=x;i++)
    {
        f=f*i;
    }
    cout<<"Factorial of "<<x<<" is "<<f<<endl;
}

int main()
{
    int i, N;
    cout<<"Enter number to find factorial:"<<endl;
    cin>>N;
    for(i=1;i<=N;i++)
    {
        factorial(i);
    }
    getch();
    return 0;
}
```

Output

Destructor in c++

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

class Marks
{
public:
  int maths;
  int science;

  //constructor
  Marks()
  {
    cout << "Inside Constructor"<<endl;
    cout << "Object created"<<endl;
  }

  //Destructor
  ~Marks()
  {
    cout << "Inside Destructor"<<endl;
    cout << "Object destructed"<<endl;
  }
};

int main( )
{
  Marks m1;
```
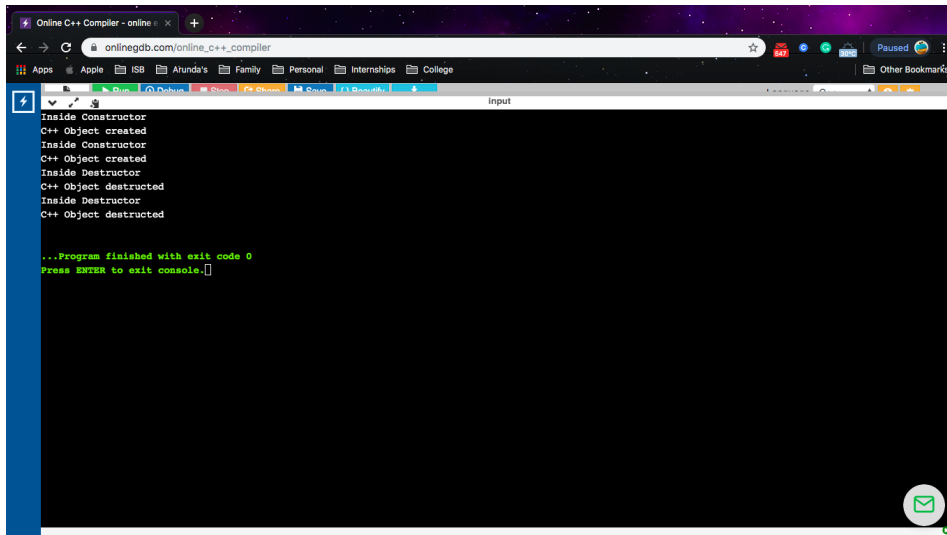
```
    Marks m2;
    return 0;
}
```

Output