```
#include <iostream>
using namespace std;
inline int Max(int x, int y) {
  return (x > y)? x : y;
}
int main() {
  cout << "Max (20,10): " << Max(20,10) << endl;
  cout << "Max (0,200): " << Max(0,200) << endl;
  cout << "Max (100,1010): " << Max(100,1010) << endl;
  return 0;
}
//Inline and non-inline member functions
#include <iostream>
using namespace std;
class example
{
   int a,b;
   public:
   void input()//Member functions which are defined inside the class are by default inline in nature
   {
      cout<<"\nEnter a and b:";
      cin>>a>>b;
   }
   void output();//By default ouput() is non-inline, but we can make it inline by using inline keyword
when it will be defined outside
};
inline void example::output()
{
   cout<<"\nValues of a and b are:"<<a<<" "<<b;
}
int main() {
  example obj;
  obj.input();
  obj.output();
  return 0;
}
//Static data member(or class variable/member)
#include<iostream>
using namespace std;
class item
 {   static int count;
     int number;
    public:
    void getdata(int d)
       {
         number = d;
         count++;
       }
```

```
  void getcount()
   {
      cout<<count;
   }
};
int item :: count;  // definition of static data member
 int main(){
 item a,b,c;
 a.getcount();
 b.getcount();
 c.getcount();

 a.getdata(100);
 b.getdata(200);
 c.getdata(300);

cout<<"\nAfter reading:";
 a.getcount();
 b.getcount();
 c.getcount();
}
//static member function
#include<iostream>
using namespace std;
class test
{
int code;
static int count;
public:
      void setcode()
      {
        code= ++count;
      }
      void showcode()
       {
       cout<<"Code: "<<code<<endl;
         }
       static void showcount()
         {
       cout<<"Count: "<<count<<endl;
         }
  };
  int test :: count;
   int main()
   {
     test t1,t2;
     t1.setcode();
     t2.setcode();
     test :: showcount();
```

```
        test t3;
    t3.setcode();

        test:: showcount();
    t1.showcode();
    t2.showcode();
    t3.showcode();
}
//Call by value
 #include<iostream>
  using namespace std;
  void add(int,int);//Function prototype or declaration
  int main()
  {
    int a,b;
    cout<<"\nEnter values of a and b:";
    cin>>a>>b;
    add(a,b);//Call by value[a,b are actual arguments][Default parameter passing technique], Function
calling
    return 0;
  }
  //Function definition
  void add(int x,int y)//[x,y are formal arguments[Duplicate copies of a and b]
  {
     cout<<"\nSum is:"<<x+y;
  }
```