static member functions
==========
1. In C++, just like we can have static data members in a class, similarly we also can have static member functions.

2. A static member function is a member function which is declared using the keyword static inside the class body.

3. We should declared a member as static if it is only accessing static data member of the class in its body. For example in the previous program the member function showcount() should be declared as static since it is only accessing the static data member count in its body.
static void showcount();

4. We must remember that the static keyword should only be used while declaring the static member and not while defining it. So, the definition of showcount() will be

```
void Emp::showcount()
{
   cout<<"Total emp working are:"<<count<<endl;
}
```

5. If a member function has been declared as static then there is an important restriction we must follow and the restriction is that **A STATIC MEMBER FUNCTION CAN'T ACCESS NON-STATIC DATA IN ITS BODY.**
This is because non-static data members accessed in a member function always belong to the calling object but a static member function never has any calling object since it is always called using class name. The general syntax of calling a static member function is

className::<static member fun name>();

For example: **Emp::showcount();**
6. C++ allows us to declare every member function as static except its constructor and desctructor. This is because constructor and desctructor are tightly bound to objects while a static member function is totally object independent member function. Moreover the main role of constructor is to initialize non-static member of the class and similarly a destructor also operates on non-static members. But if we declare them as static member function then we will not be able to perform their activites. **Thus constructor and destructor are always declared as non-static member function**

Version 3

```
#include <iostream>
#include <string.h>
using namespace std;
class Emp
{
   int age;
   char name[20];
   static int count;
public:
   Emp(int,char *);
   void show();
   static void showcount();
   ~Emp();
};
int Emp::count;
Emp::Emp(int a,char *p)
{
   age=a;
   strcpy(name,p);
   ++count;
}
void Emp::show()
{
   cout<<age<<","<<name<<endl;
}
void Emp::showcount()
{
   cout<<"Total emp working are:"<<count<<endl;
}
Emp::~Emp()
{
   --count;
}
```

```
int main()
{
  {
   Emp::showcount(); —> 0
   Emp E(21,"Amit");
   Emp F(24,"Ravi");
   Emp G(22,"Nitin");
   E.show();
   F.show();
   G.show();
   Emp::showcount(); —> 3
   {
      Emp X(23,"Sooraj");
      Emp Y(22,"Kiran");
      X.show();
      Y.show();
      Emp::showcount(); —> 5
   }
   Emp::showcount(); —> 3
}
Emp::showcount(); —> 0
return 0;
  }
```

Modify the previous code , so that now your program does the following:

1. Provide salary to every emp
2. Modify the body of CONSTRUCTOR to initialize salary with the argument passed
3. Provide a member function in your class which display the average salary.