Const and Static data members

Const data members

- const data members can be initialized in constructor with initialization list (pure initialization)
- const data members can not be changed after initialization.

Red coloured snippets gives error

```
class Sample{
private : const int k;
public:
Sample():k(10){}
};
```

```
class Sample{
private :
const int k;
public:
Sample(){
k=10}
};
```

```
class Sample{
private :
 const int k =10;
public:
Sample(){  }
};
```

Const Member functions

- const member function makes entire **invoking object read only inside it** and does not allow any changes to any data member.

```
#include<iostream>
using namespace std;
    class Complex{
        private: int i,j;
        public:
            void Accept() {
            cout<<"\nEnter real & img\n";
                cin>>i>>j; }

            void Display() const{
                cout<<"\n"<<i<<" "<<j;
            //  i= i+1;  //Generate  error
                } };
                int main(){
                    Complex c1;
                    c1.Accept();
                    c1.Display();
                return 0;}
```

Const Objects

- In C++, We can make objects constant using const keyword.

Invoking object	Member Function Type	Legal/Illegal
const object	const member function	LEGAL
const object	Non-const member function	ILLEGAL
Non-const object	const member function	LEGEL
Non-const object	Non const member function	LEGAL

Const Objects and Const Member Functions


```
#include<iostream>
using namespace std;
class Complex{
        private:
        int i,j;
        public:
void Accept() {
cout<<"\nEnter real and img part";
        cin>>i>>j;
 }
    void Display() const{
        cout<<"\n"<<i<<" "<<j;
//  i= i+1;  //Generate  error
    } };

    int main(){
        Complex c1;
        c1.Accept();
        c1.Display();
    return 0;}
```

Static data members

- Static data members gets initialized once and resides in data segment.
- Static data members are called as Class Members & not instance members
- Static data members are shared by all the instances or objects of class.
- Static data members does not contributes to size of object
- Static data members needs to be initialized outside class

Static Member Function

- Static member functions can be invoked without object i.e. using class name only using :: (Scope resolution operator)
- Static members can access static data members only but not the instance data members
- Static member functions does not receive **this pointer** as they not invoked by object.

Static Data Members & Member Functions


```
#include<iostream>
using namespace std;
class Complex{
    private:
    int i,j;
    static  int count;
    public:
Complex():i(10),j(20) {
    count++; }
static int GetCount(){
    //i = i+1; // error
    return count;
} };
int Complex::count = 0;
//Init outside class(REQUIRED)
int main(){
    Complex c1,c2;
    int n = Complex::GetCount();
    cout<<"\n Count="<<n;
return 0;}
```

this pointer

- this pointer represents invoking object inside instance member functions
- Compiler automatically create and pass this pointer in instance member functions
- Being const pointer to object, value can not be assigned to this ptr.

  Ex. this = NULL; // not permitted (lvalue required)
- Static data members does not receive this pointer.

Declaration for member function :

Complex *const this = <address>

Declaration for const member function :

const Complex *const this=<address>

Thank You

Look but don't touch!!!!

…….const (read only)