- **22. Приятелски класове и приятелски функции. Статични членове на клас.**

# Static Members

- Static member variable:
  - One instance of variable for the entire class
  - Shared by all objects of the class
- Static member function:
  - Can be used to access static member variables
  - Can be called before any class objects are created

# Static Member Variables

1) Must be declared in class with keyword **`static`**:

```
class IntVal
{
 public:
    IntVal(int val = 0)
    { value = val; valCount++ }
    int getVal();
    void setVal(int);
  private:
    int value;
    static int valCount;
};
```

# Static Member Variables

2) Must be defined outside of the class:

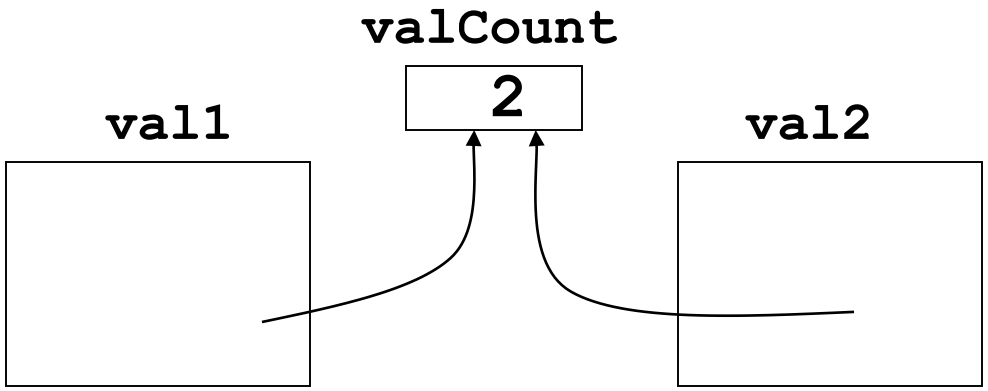
```
class IntVal
{
    //In-class declaration
    static int valCount;
    //Other members not shown
};

//Definition outside of class
int IntVal::valCount = 0;
```

# Static Member Variables

3) Can be accessed or modified by any object of the class: Modifications by one object are visible to all objects of the class:

```
IntVal val1, val2;
```



# Static Member Functions

1)Declared with **`static`** before return type:

```
class IntVal
{ public:
     static int getValCount()
    { return valCount; }
  private:
     int value;
     static int valCount;
};
```

# Static Member Functions

2) Can be called independently of class objects, through the class name:

```
cout << IntVal::getValCount();
```

3) Because of item 2 above, the **`this`** pointer cannot be used

4) Can be called before any objects of the class have been created

5) Used mostly to manipulate static member variables of the class

# Friends of Classes

- Friend function: a function that is not a member of a class, but has access to private members of the class
- A friend function can be a stand-alone function or a member function of another class
- It is declared a friend of a class with the **`friend`** keyword in the function prototype

# Friend Function Declarations

1) Friend function may be a stand-alone function:

```
class aClass
{
  private:
     int x;
     friend void fSet(aClass &c, int a);
};

void fSet(aClass &c, int a)
{
    c.x = a;
}
```

# Friend Function Declarations

2) Friend function may be a member of another class:

```
class aClass
{ private:
    int x;
    friend void OtherClass::fSet
                        (aClass &c, int
a);
};
class OtherClass
{ public:
     void fSet(aClass &c, int a)
     { c.x = a; }
};
```

# Friend Class Declaration

3) An entire class can be declared a friend of a class:

```
class aClass
{private:
   int x;
   friend class frClass;
};
class frClass
{public:
   void fSet(aClass &c,int a){c.x = a;}
   int fGet(aClass c){return c.x;}
};
```

# Friend Class Declaration

- If **`frClass`** is a friend of **`aClass`**, then all member functions of **`frClass`** have unrestricted access to all members of **`aClass`**, including the private members.

- In general, restrict the property of Friendship to only those functions that must have access to the private members of a class.