5 Objects and Classes in C++

- Class declaration similar to a structure declaration in C
- Access to members of an object (instance variables and methods) with the operators `.` or `->`, like the access to structure components
- Example:

```
// Class counter
class Counter
{
  private:
    int value;
  public:
    void incr() { value++; }
    void decr() { value--; }
    int get_value() { return value; }
};
```


6 Methods in C++

- Definition within a class declaration:
  - ➤ method is handled as ***inline*** function
- Definition separate from the class declaration
  - ➤ assignment to class with the ***scope*** operator `::`
  - ➤ method invocations are handled like normal function calls
- Example:

```
class Counter {
  private:
    int value;
  public:
    void incr(); void decr(); int get_value();
};

void Counter::incr()         { value++; }
void Counter::decr()         { value--; }
int Counter::get_value()     { return value; }
```


7 Instantiation in C++

- Instantiation of Objects either
  - ➤ statically at compile time, or
  - ➤ dynamically during run time

★ **Static Instantiation**

- By object definition
- Example:

```
void main()
{
  Counter c1;          // object c1 of class Counter
  Counter *pc1;        // pointer to an object of class Counter
  ...
}
```


7 Instantiation in C++ (2)

★ **Dynamic Instantiation**

- C++ operators **`new`** and **`delete`**
- Example:

```
class Counter
{ ... };

void main()
{
  Counter c1;              // create object c1 statically
  Counter *pc1;            // pointer to an object of class Counter
  ...
  pc1 = new Counter;
  pc1->incr();
  c1.incr();
  ...
  delete pc1;
  ...
}
```