Class (1A)

Please send corrections (or suggestions) to youngwlim@hotmail.com.

This document was produced by using OpenOffice.

Class Definition

```
class Ccircle {
public:
  int r;

  Ccircle ()          { r = 1; }
  Ccircle (int x)     { r = x; }

  void      setR (int x) { r = x; }
  int       getR ()    { return r; }
  double area () ;
} ;
```

A Member

Constructor functions

Member functions

member functions defined outside the class definition

```
double Ccircle::area () {
   return 3.14*r*r;
}
```

Scope Operator ::

Ccircle::

Creating Objects

```
class Ccircle {
public:
  int r;

  Ccircle ()        { r = 1; }
  Ccircle (int x)   { r = x; }

  void      setR (int x) { r = x; }
  int       getR ()   { return r; }
  double area () ;
} ;
```

```
double Ccircle::area () {
   return 3.14*r*r;
}
```

```
void main(void)  {

  Ccircle C1;      default constructor is called
  Ccircle C2(10);

}
```

Ccircle C1();

object **C1**

r = 1

setR ()
getR ()
area ()

object **C2**

r = 10

setR ()
getR ()
area ()

- Member functions are called from objects
- Objects have their own member data
- So member functions access these distinct member data

Calling Member Functions

```
class Ccircle {
public:
  int r;

  Ccircle ()          { r = 1; }
  Ccircle (int x)     { r = x; }

  void      setR (int x) { r = x; }
  int       getR ()    { return r; }
  double area () ;
} ;
```

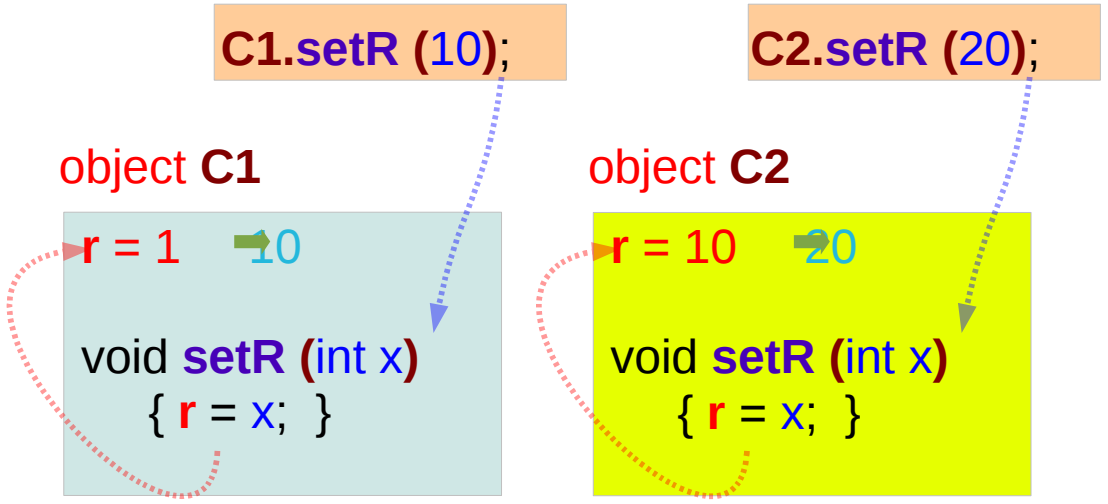
```
void main(void)  {

  Ccircle C1;      default constructor is called
  Ccircle C2(10);

  C1.setR (10);
  C2.setR (20);

}
```

```
double Ccircle::area () {
   return 3.14*r*r;
}
```



Member Functions : called from objects

```
class Ccircle {
public:
  int r;

  Ccircle ()          { r = 1; }
  Ccircle (int x)     { r = x; }

  void      setR (int x) { r = x; }
  int       getR ()    { return r; }
  double area () ;
} ;
```

```
void main(void)  {

  Ccircle C1;      default constructor is called
  Ccircle C2(10);

  C1.setR (10);
  C2.setR (20);

  C1.area () ;
  C2.area () ;

}
```

```
double Ccircle::area () {
   return 3.14*r*r;
}
```

- Member functions are called from objects
- Objects have their own member data
- So member functions access these distinct member data

Objects and Member Function Calls

```
Ccircle C1;
C1.setR (10);
```

object **C1**

```
r = 10

setR ()
getR ()
area ()
```

```
C1.area ();
```

➡ 3.14*10^2

object **C1**

```
r = 10

double area ()
  { return 3.14*r*r; }
```

```
Ccircle C2(10);
C2.setR (20);
```

object **C2**

```
r = 20

setR ()
getR ()
area ()
```

```
C2.area ();
```

➡ 3.14*20^2

object **C2**

```
r = 20

double area ()
  { return 3.14*r*r; }
```

Member Function Call

```
class Ccircle {
public:
  int r;

  Ccircle ()          { r = 1; }
  Ccircle (int x)     { r = x; }

  void      setR (int x) { r = x; }
  int       getR ()    { return r; }
  double    area () ;
} ;
```

```
void main(void)  {

  Ccircle C1;
  Ccircle C2(10);

  C1.setR (10);
  C2.setR (20);

  C1.area () ;
  C2.area () ;

}
```

possible implementation :

```
Ccircle::setR (&C1, 10);
```

```
void Ccircle::setR (Ccircle *this, int x)
{
    this->r = x;
}
```

```
r = x;
```

passing a pointer hidden to a programmer

Access Specifier

```
class Ccircle {
public:
  int r;

  Ccircle ()          { r = 1; }
  Ccircle (int x)     { r = x; }

  void      setR (int x) { r = x; }
  int       getR ()    { return r; }
  double area () ;
} ;
```

```
class Ccircle {
  int r;          default private:

public:
  Ccircle ()          { r = 1; }
  Ccircle (int x)     { r = x; }

  void      setR (int x) { r = x; }
  int       getR ()    { return r; }
  double area () ;
} ;
```

```
void main(void)  {

  Ccircle C1;
  Ccircle C2(10);

  C1.setR (10);
  C2.setR (20);
```

```
  C1.r = 13;
  C2.r = 24;


}
```

Member Function Definition within a class

```
int func1( ) {
      mem2 = 10;
       func2 ();
      mem3 = 10;
       func3 ();
}
```

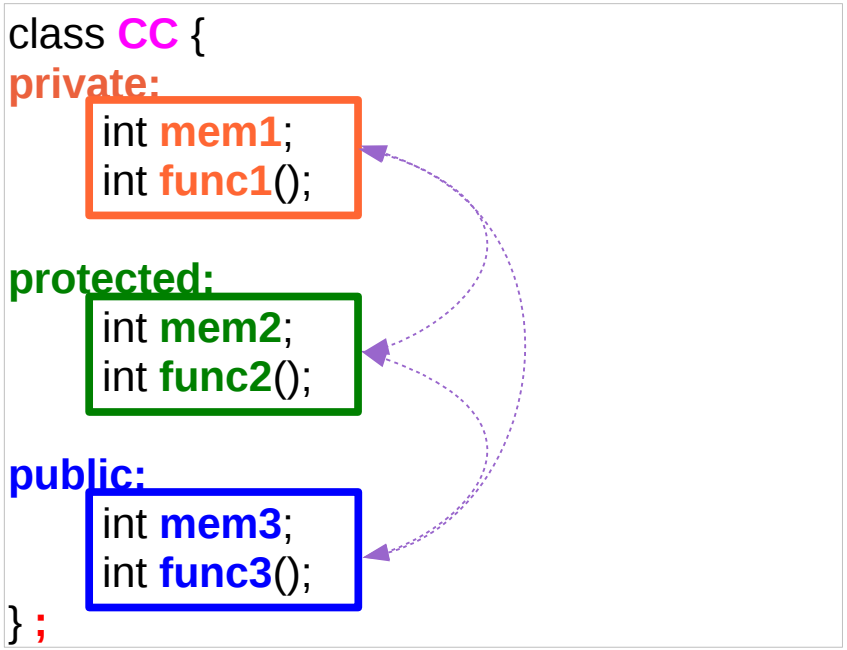
member functions of the same class

```
int func2( ) {
      mem1 = 10;
       func1 ();
      mem3 = 10;
       func3 ();
}
```

member functions of the same class

```
int func3( ) {
      mem1 = 10;
       func1 ();
      mem2 = 10;
       func2 ();
}
```

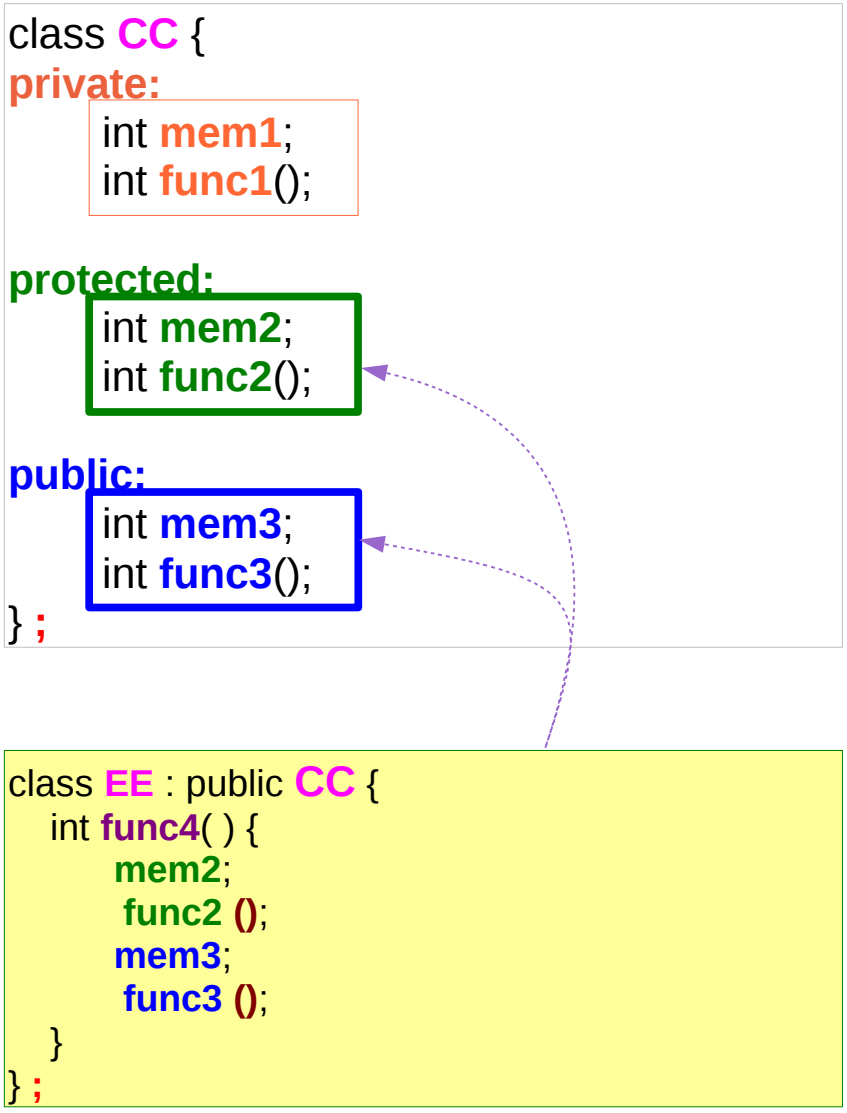
member functions of the same class



Each members can be accessed by other members of the same class

Member Function Definition within a derived class

The members of a derived class can access public and protected members of the base class



Member Function Call from objects

```
void main(void)  {                    the main function
  CC C1;

  C1.mem3;
  C1.func3 ();
}
```

```
int foo(CC *X)  {                     C-style functions

  X->mem3;
  X->func3 ();
}
```

```
class DD {                            member functions of
    int faa(CC *Y) {                  other classes
          Y->mem3;
          Y->func3 ();
    }
} ;
```

```
class CC {
private:
        int mem1;
        int func1();

protected:
        int mem2;
        int func2();

public:
        int mem3;
        int func3();
} ;
```

Only public members can be accessed

Class Structure

Class Structure

References

[1] W Savitch, “Absolute C++”
[2] P.S. Wang, “Standard C++ with objected-oriented programming”
[3] http://www.cplusplus.com