Operator Overloading — 重載：相同函式名稱 不同參數形式

運算元：+ - * / , = , == , > , >= , [] , +- , --

⇔ 函式

ex: int a, b, c;

a = b = c; ← Assignment operator

int b.operator=(int c); c：輸入參數

輸出型態：int　函式名：operator=　自己本身："b"

把 c 的資料複製到 b

int a.operator=(int); b

輸出：int　a 自身

ex: 自訂類別

MyClass A, B, C;

A = B + C;

MyClass B.operator+(MyClass C); C

函式名：operator+

(B和C相加的結果)

新的暫存的 MyClass 物件

MyClass A.operator=(MyClass);

輸出：MyClass　函式名：operator=　輸入：MyClass

2014-May-13

資工一甲

```
class pos // 座标類別
{private:
   int x, y; //2D 座标
 public:
    pos()
    { x=y=0;
    }
    pos(int a, int b)
    { x=a;
      y=b;
    }
    pos(pos& p) // 複製建構元
    { x=p.x;
      y=p.y;
    }
    pos operator=(pos r)
    { x=r.x;
      y=r.y;
      return *this;
    }
    pos operator+=(pos rhs) // d+=C;
    {  x=x+rhs.x;
       y=y+rhs.y;
       return *this;
    }
```

```
int main()
{
   pos a; //(0,0)
   pos b(8,6); //(8,6)
   pos c(b); //(8,6)
   a=c;
   pos d(5,2);
   d+=c; //(13,8)
   a=(d+c); //(21,14)
}
```

伝参考

Copy Constructor

輸入:另一座标物件

輸出自身物件

C++关键字

一個座标物件

*this 型態 pos

#122

x

y

pos

*this=#122

自身的座标物件

指向自身的指标 (編譯產生)

right hand side 右手边

輸出自身物件

C:(8,6)

(13,8)

(8,6)

<=> pos d.operator+(pos)

輸出相加暫存的座标

```
pos operator+(pos rhs)
{
   pos ans(rhs); // 複製建構元
   ans+=(*this); // d:(13,8)
   return ans;
}
```

Operator ⟺ function

程式撰寫次序

- ---- Copy Constructor
- ---- "=" operator
- ---- "+=" operator / "-="
- ---- "+" operator / "-"

```
pos operator=(pos r)   // A=B
{                      ⟺ A.operator=(B)
   x = r.x;
   y = r.y;
   return *this;
}
```

傳值參數傳遞（作）

B 複製一份當 r

傳值

把自身物件複製一份當作輸出參數

需要額外記憶体
執行效率較差

```
⇒ pos& operator=(pos& r)
{  傳參考       傳參考
   r.x = 0;
   ____
   ____
   return *this;
}
```

r 即 B 本身

r 是 B 的別名

```
⇒ pos& operator=(const pos& r)
{
   r.x = 0
}
```

常數 – const；座標 – pos；傳參數 – &

~~r.x = 0~~

```
pos&  operator=(pos& rhs)
{
    if (this == &rhs)
        return *this;
    ________
    ________
    ________
}
```

const

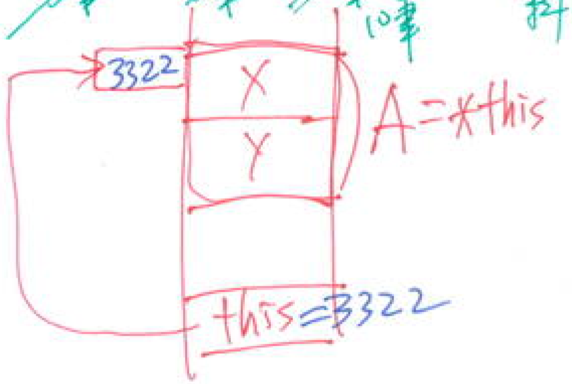
A

3322

3322

// A=A   A=B

5筆 5筆 5筆 10筆 = 10個資料



const 常数 constant

用法

```
class A
{ private:
    int y;
  public:
    A(): y(0) { }
    int f1(int x) const
    { x++;
      y++;
    } return y;
    int f3(int const x)
    { x++;
      y++;
      return y;
    }
    void f4(int const *x)
    { x++;
      (*x)++;
    }
    void f5(int const &x)
    { x++;
    }
    void f6(int * const x)
    { x++;
      (*x)++;
    }
    void f8(int const * const x)
    { x++;
      (*x)++;
    }
```

常数函数 不能改变类别中的成员资料

整数 常数的

整数 常数 传参考代表

整数 常数的 指标指向

指标 常数的

整数 常数 指标指向 常数