

運算子多載

Operator Overloading for C++

Made By Chocomint

What is Operator?

- 運算子是一個符號，它可以告訴編譯器執行指定的數學關係或邏輯運算並產生最終結果。
- C++中常見的運算子有以下幾種：

+ - * / =

All operator in C++ (1)

優先級	運算子	優先級	運算子	優先級	運算子	優先級	運算子
1	::	3	!	4	.*	9	<
2	++(尾)		~	5	->*		<=
	--(尾)		(type)		*		>
	()		*		/		>=
	[]		&	6	%	10	==
	.		sizeof		+(calc)		!=
	->		new	7	-	11	&
	++		new[]		<<	12	^
3	--		delete	8	>>	13	
	+		delete[]		<=>	14	&&
	-					15	

All operator in C++ (2)

優先級	運算子	優先級	運算子
16	A?B:C	17	&=
17	=		^=
	+=		=
	-=	18	throw
	*=	19	,
	/=		
	%=		
	<<=		
	>>=		

How to Overloading Operator?

➡ 基本型態：

```
[return type] [class]::operator[op]((const) [data type] (&)[data name])
```

➡ Eg.

```
double hello::operator+(const int &ref)
```

➡ ※ “&” 在函數中表 “參考(reference)” 之義

Binary Operator (雙元運算子)

➤ 運算子兩側都有接變數，即稱「雙元運算子」

➤ Eg.

□ $x + y$

□ $x \% y$

□ $x == y$

□ $x < y$

□ $x << y$

□ $x || y$

Overloading for General Binary Operator

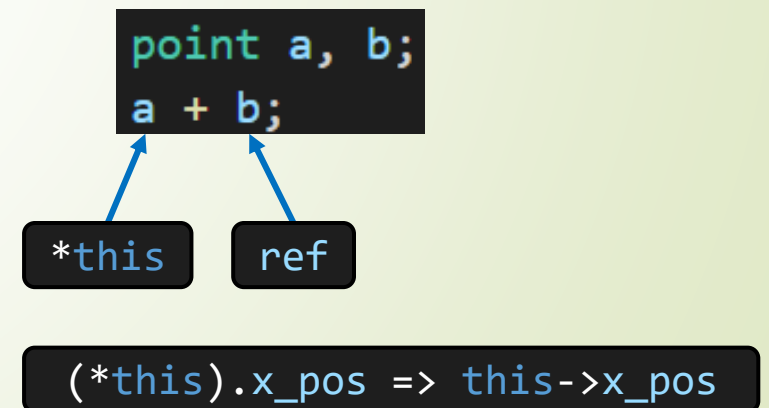
```
class point
{
private:
    int x_pos, y_pos;

public:
    point operator+(const point &ref);
};
```

函數定義在 class 中

class 名稱 + ::

```
point point::operator+(const point &ref)
{
    point tmp;
    tmp.x_pos = this->x_pos + ref.x_pos;
    tmp.y_pos = this->y_pos + ref.y_pos;
    return tmp;
}
```



Overloading for Assignment Operator

```
void point::operator=(const point &ref)
{
    this->x_pos = ref.x_pos;
    this->y_pos = ref.y_pos;
}
```

賦值運算子主要是
對 this 進行修改

```
point a, b, c;
c = a + b;
```


Unary Operator (單元運算子)

- 只有接一個變數的運算子即稱作「單元運算子」
- Eg.
 - ❑ 負號
 - ❑ $x++$ 、 $++x$
 - ❑ $!$ 、 \sim (NOT)