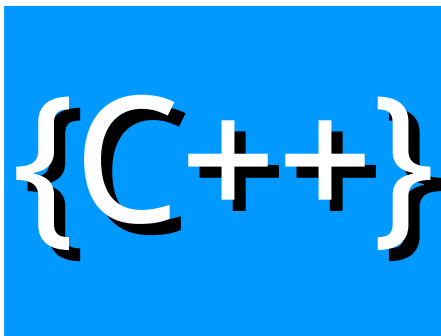




Inheritance

Week 6



Yang-Cheng Chang
Yuan-Ze University
yczhang@saturn.yzu.edu.tw



Assignment 6

- 撰寫一個簡單的怪物對戰程式
 - 怪物具有目前生命值、最大生命值、攻擊力與防禦力
 - 怪物之間的對戰，採取回合制，每一回合互相攻擊一次
 - 怪物的基本攻擊公式
$$\text{敵對怪物目前生命值} = \text{敵對怪物目前生命值} - (\text{攻擊力} - \text{敵對怪物的防禦力})$$
 - 每種怪物具有特殊攻擊模式



class Monster

- 撰寫一個基礎類別 `Monster`，具有以下成員
 - 屬性
 - `HP` 代表目前生命值
 - `MaxHP` 代表最大生命值
 - `Attack` 代表攻擊力
 - `Defense` 代表防禦力
 - 存取函式
 - `setHP()`, `getHP()`, `getMaxHP()`
 - `getAttack()`, `getDefense()`
 - 攻擊與狀態函式
 - 所有繼承 `Monster` 的類別可以重新實現 `showStats()`, `attack()` 兩個函式



class Monster

- 攻擊與狀態函式說明

attack() 進行攻擊計算

showStats() 顯示狀態，輸出的格式為
怪物的名稱 (HP / MaxHP)，例如
Unicorn(68 / 500)

- 建構子必須使用參數來初始化成員屬性

```
1 class A {  
2     private:  
3         int size;  
4     public:  
5         A(int s): size(s){}  
6 };
```



使用基礎型別的建構函式來初始化類別

```
1 class A {  
2     private:  
3         int size;  
4     public:  
5         A(int s): size(s){}  
6 };  
7  
8 class B : public A {  
9     private:  
10        int count;  
11    public:  
12        B(int cnt): A(10), count(cnt) {}  
13  
14 };
```



class Dragon

- 撰寫類別 Dragon ，繼承 class Monster ，具有以下成員
 - Rate 用來計算 Dragon 的附加攻擊比率
 - showStats(), attack()
- Dragon 的特殊攻擊模式為附加攻擊
 - 附加攻擊計算公式
隨機一個數值 (1-10) ，再乘上 Rate 後為附加攻擊
 - 攻擊計算公式
$$\text{敵對怪物目前生命值} = \text{敵對怪物目前生命值} - (\text{攻擊力} - \text{敵對怪物的防禦力}) - \text{附加攻擊}$$



class Unicorn

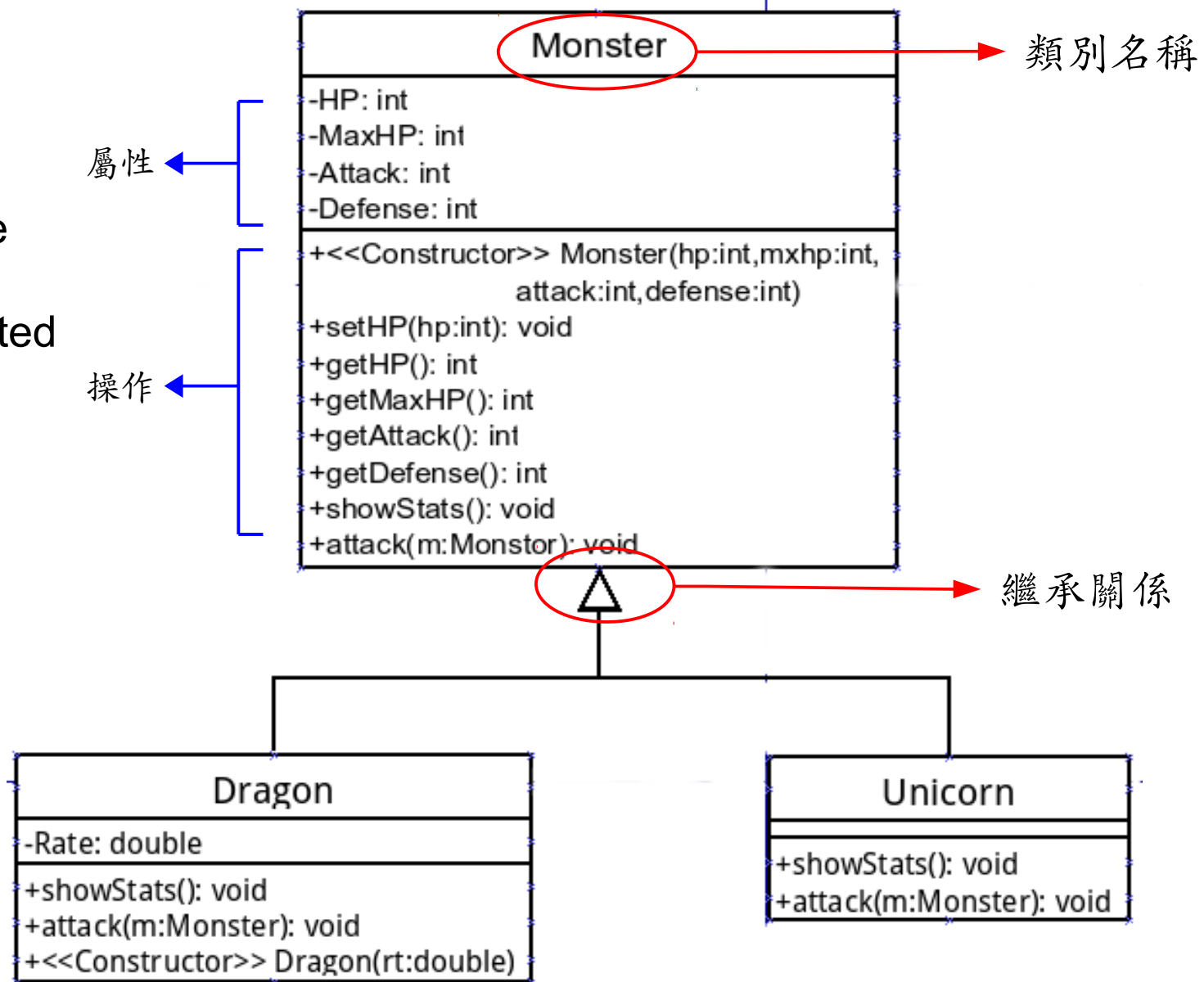
- 撰寫類別 Unicorn ，繼承 class Monster ，具有以下成員
 - showStats(), attack()
- Unicorn 的特殊攻擊模式
 - 進行基本攻擊後，具有 1/4 的機率再次進行基本攻擊
 - 用查表的方式來判定機率攻擊
 - 初始化一個具有 4 個元素的 int 矩陣，其中某個元素的值為 1，其他都設定為 0

		0	1	2	3
index					
int 型別的陣列		0	0	1	0

- 隨機產生一個介於 0-3 的數字，然後以這個數字來檢索陣列
若檢索到的陣列元素值為 1，則進行再次的基本攻擊

UML 繼承圖

- 代表 private
- + 代表 public
- # 代表 protected





主程式

- 修改主程式，隨機決定 m1 或 m2 先發動攻擊

```
1 #include <iostream>
2 #include "monster.h"
3
4 int main(int argc, const char *argv[])
5 {
6     Dragon m1(0.6);
7     Unicorn m2;
8
9     int cnt = 1;
10    while(true){
11        std::cout << "Round " << cnt << std::endl;
12        m1.attack(m2);
13        m2.showStats();
14        if( m2.getHP() <= 0 ){
15            std::cout << ">>> Unicorn died, Dragon win! <<<" << std::endl;
16            m1.showStats();
17            break;
18        }
19        m2.attack(m1);
20        m1.showStats();
21        if( m1.getHP() <= 0 ){
22            std::cout << ">>> Dragon died, Unicorn win <<<" << std::endl;
23            m2.showStats();
24            break;
25        }
26        std::cout << "=====" << std::endl;
27        cnt++;
28    }
29    return 0;
30 }
```