L’operatore per stampare (con ostream) non lo possiamo mettere nella classe

* Lo metto nello stesso .cpp della classe, ma non nella classe (e non ci metto nomeclasse:: )
* Per fargli vedere le variabili della classe, uso   
  friend std::ostream& operator<< bla bla (vedere il file arma.cpp in l11)

Classe astratta: non può essere istanziata, le sue classi figlie devono avere necessariamente tutti i metodi che ha lei (e devono essere implementati, mentre lei li dichiara soltanto)

In C++ si fa una classe normale, se almeno un metodo è virtual allora la classe diventa astratta (es: virtual void lol())

Extends in C++: class guerriero : personaggio{…} (posso fare anche public personaggio o private personaggio, private significa che dall’esterno non si può vedere che guerriero è un personaggio, ma solo da dentro guerriero)

Polimorfismo: in questo caso il guerriero è contemporaneamente guerriero e personaggio

**NB:** non posso fare

PG a = Warrior(..);

a.attributoDaPG = qualcosa;

perché nella stack non ci entrano le variabili

//VEDERE SPIEGAZIONE CIRCA AL MINUTO 29

**Binding dinamico:** se ho due funzioni in una classe e nella madre, va nella “parte più bassa della gerarchia” a prenderla (es: da guerriero, non personaggio).

* Specifico override in guerriero
* Ma serve anche specificare virtual al metodo in personaggio

**NB:** posso comunque fare solo override ma non sarà binding dinamico

**Protected:** visibili solo da classe e sottoclassi, non dall’esterno

In caso di sottoclassi i distruttori vengono cominciati ad essere chiamati dal basso, ogni distrutto deve distruggere le variabili della propria sottoclasse

Final class non può essere estesa

Final funzione non può essere overridata? (1 ora e 10 registrazione)

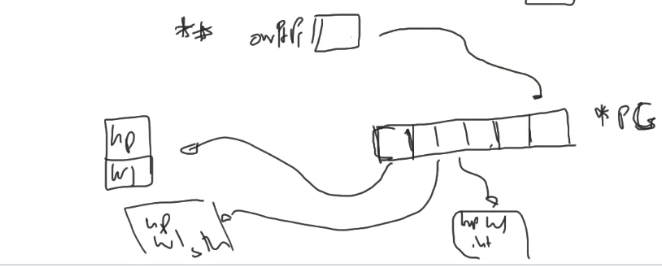
**NB:** char e int possono essere confuse, se io una classe ho f(int a);

e in una sottoclasse f(char b)

E richiamo f(5); viene comunque richiamata f(char b);

Come faccio se voglio un array di varie sottoclassi di personaggio?

Faccio un array di puntatori, ognuno di tipo “personaggio”

PG \*\*p = new \*PG[n];

**NB:** dovrò fare il delete di ogni elemento dell’array

**I CAST (iniziato alle 17.03)**

* Dynamic cast: Guerriero \*g = dynamic\_cast<Guerriero\*>(\*puntatoreAPersonaggio)  
  Più veloce ma non fa controlli
* Static cast: : Guerriero \*g = static\_cast<Guerriero\*>(\*puntatoreAPersonaggio)   
  Questo controlla prima se si può fare il cast

**NB:** nella gerarchia posso fare solo su e giù, non sinistra e destra.

**Copyconstructor**

Se faccio

Guerriero g1;

Guerriero g2 = Guerriero(b);

g1 = g2; -> fa una copia di g2 e la chiama g1;

-> in questo modo è diverso, qui fa il copy assignment

-OPPURE-

Guerriero g1 = Guerriero();

Guerriero g2 = Guerriero(g1);

* In questo caso devo fare un costruttore che è adibito

PROBLEMA: se g1 punta a spada1, anche g2 punterà a spada1. Se g1 distrugge la spada, verrà distrutta anche per g2.

SOLUZIONE:

Personaggio(const Personaggio& pg){

//Copia i vari valori da pg a this

If(ha un’arma o qualsiasi altro oggetto collegato){

Equip = new Arma(…);

}

}