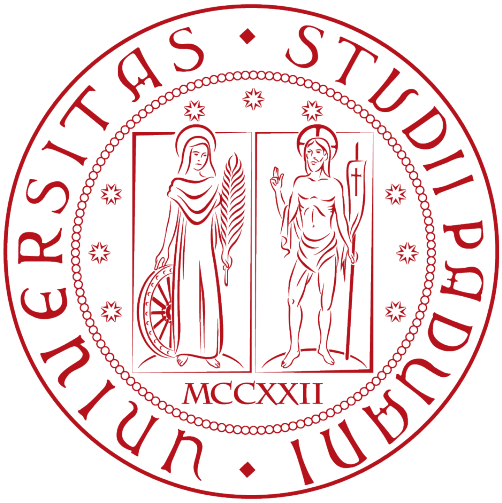
UNIVERSITÀ DEGLI STUDI DI PADOVA

SCUOLA DI SCIENZE



CORSO DI LAUREA IN INFORMATICA

PAO

QUICK REFERENCE

Riassunto Programmazione ad Oggetti

Federico Silvio Busetto Matricola:1026925

ANNO ACCADEMICO 2014-2015

Indice

Elenco delle figure

Capitolo 1. Introduzione

Queste dispense sono un riassunto super condensato del libro *Programmazione ad Oggetti di F. Ranzato* Negli esempi si useranno principalmente le classi indicate nel libro, soprattutto le classi orario, come classe di un tipo concreto (ma rimpiazzabile da qualunque altro a scelta).

Capitolo 2. Classi e Oggetti

Le *struct* al contrario delle classi, permettono di usare la rappresentazione interna dell'ADT.

2.1 Modularizzazione

Classi:

- Definizione di Interfaccia.h, il file di header con tutte le dichiarazioni dell'ADT (campi dato e metodi)
- Implementazione dei suoi metodi nel file .cpp
- in un terzo file, ci va il main

Istruzioni per modularizzazione:

- *g++ -c orario.cpp* compila orario.cpp
- *g++ -c main.cpp* compila il main
- *g++ main.o orario.o* linka i binari

L'information hiding è garantito dal fatto che all'utente verrà consegnato solo il compilato orario.o

2.2 Costruttori e Assegnazione

2.2.1 Costruttore standard

Il costruttore standard è reso disponibile dal linguaggio, ed è un costruttore di default (senza parametri)
Comportamento: Alloca la memoria e lascia i valori dei tipi primitivi e derivati (puntatori, riferimenti e array) indefiniti. Per i tipi classe richiama il costruttore di default dell'oggetto, campo dato per campo dato.
Esempio: Per costruire un Dataora (oggetto composto da due sotto oggetti, una data e un orario), il costruttore standard di Dataora invocherà il costruttore di default di data e il costruttore di default di orario, standard oppure ridefiniti.
NOTA: Nel costruttore di default, gli oggetti di altre classi possono essere costruiti richiamando il loro costruttore di default, come avviene a pag. 50 del libro dove il costruttore di default della classe telefonata, costruisce di default due oggetti di tipo orario: *telefonata::telefonata(text){numero = 0;}*

2.2.2 Costruttore di copia

Il costruttore di copia ha firma orario(const orario&){}, è riconoscibile in quattro casi:

- orario pippo = o; dove o è un oggetto di tipo orario precedentemente creato con l'istruzione orario o; In questo primo caso abbiamo una nuova variabile che viene dichiarata, seguita da un assegnazione. La combinazione tipo, nomevariabile, assegnazione, indica che si tratta in realtà di un costruttore di copia
- orario pippo(mezzanotte); se si dichiara una nuova variabile di tipo e gli si passa un altro oggetto dello stesso tipo, non è un costruttore ad un parametro, bensì un costruttore di copia.
- Nel caso di passaggio di parametri per valori in funzioni
- Nel caso di ritorno per valore in funzioni

Comportamento: Copia campo per campo.
Per approfondire: `http://www.tutorialspoint.com/cplusplus/cpp_copy_constructor.htm`

2.2.3 Assegnazione

L'assegnazione è data dal simbolo = che è un operatore ridefinibile con firma *C& operator=(const C&)*.
Comportamento: L'assegnazione standard tra due oggetti, assegna i valori membro a membro, invocando l'assegnazione standard o ridefinita del tipo. E' pericoloso in quanto può generare interferenze e condivisione non voluta di memoria se l'assegnazione tra campi riguarda puntatori.

2.2.4 Costruttore ad un parametro

Il costruttore con un solo parametro agisce anche come convertitore di tipo (ciò può essere bloccato usando la keyword *explicit*)
Esempio: il costruttore *orario(int){}* genera una conversione implicita dal tipo int al tipo orario (ma non il viceversa).
Sarebbe come ridefinire *operator int()* dichiarandolo nella parte pubblica della classe orario, generando una conversione esplicita.

2.2.5 Ridefinizione di operatori

Un operatore può essere ridefinito come metodo interno alla classe o come funzione esterna. Come metodo, ha sempre un parametro in meno rispetto a quelli richiesti, ad esempio un operatore binario come operator+, avrà un solo parametro tra parentesi, mentre un operatore unario non avrà parametri.Questo vale solo per gli operatori ridefiniti come metodi propri della classe, se invece fossero ridefiniti come funzioni esterne, allora avrebbero un parametro in più di tipo classe (quello che era l'oggetto di invocazione, viene passato per parametro).
Un operatore è ridefinibile solo se tra i parametri ha almeno un tipo definito dall'utente. Ridefinendo un operatore come funzione esterna, non si necessita dell'operatore di scoping nella definizione del metodo nel file .cpp ma l'operatore perderà l'accesso alla parte privata della classe.
NOTA: gli operatori [] e () possono essere ridefiniti solo come metodi propri.
`https://stackoverflow.com/questions/4172722/what-is-the-rule-of-three`(The Rule of Three - StackOverflow)