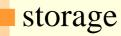
Static



• variabili static in una function

scope

• variabili static in una classe / file

AA 2001-2002

programmazione ad oggetti

.

static in function

- variabili di una function
 - storage allocato ad ogni chiamata (stack)
 - inizializzate ad ogni chiamata
- variabili static

StaticVariablesInfunctions.cpp
StaticObjectsInFunctions.cpp
StaticDestructors.cpp

- storage fisso
- inizializzate solo la prima volta
- visibilità locale ma storage globale
- oggetti, idem con attenzione ai costruttori

AA 2001-2002

programmazione ad oggetti

2

static in file

- nomi (var e function) dichiarati a livello di file sono visibili globalmente (usando dichiarazioni extern)
- un nome dichiarato static è visibile solo nel file in cui è dichiarato
- lo storage per le variabili è globale
- in C++ l'uso di file static è sconsigliato (deprecated)

 LocalExtern.cpp

AA 2001-2002

programmazione ad oggetti

3

LocalExtern2.cpp

Namespaces

- suddivisione dello spazio di nomi globale in parti separate
- altera solo la visibilità (non storage)
- sostituisce (e migliora) i file static
- si può suddividere un namespace in più namespaces (gerarchia)
- si può definire un namespace in più file

AA 2001-2002

programmazione ad oggetti

Definizione di Namespaces

```
namespace MyLib {
                              // Declarations
HEADER1.H
                                             namespace BobsSuperDuperLibrary {
                                               class Widget { /* ... */ };
class Poppit { /* ... */ };
namespace MyLib {
  extern int x;
  void f();
                                             // Too much to type! I'll alias it:
       HEADER2.H
                                             namespace Bob = BobsSuperDuperLibrary;
       #include "Header1.h"
       // Add more names to MyLib
       namespace MyLib {
         extern int y;
         void g();
         // ...
 AA 2001-2002
                               programmazione ad oggetti
```

Uso di Namespaces

scope resolution operator

namespace::internal_name

ScopeResolution.cpp

- la direttiva using
 - importa un intero namespace using namespace_name;

NamespaceInt.h NamespaceMath.h Arithmetic.cpp

- la dichiarazione using
 - importa un singolo nome da un namespace using namespace::internal_name;

UsingDeclaration.h UsingDeclaration.cpp UsingDeclaration.cpp

static in class

- storage unico per tutti gli oggetti istanziati dalla classe
 - globale rispetto a tutti gli oggetti, ma può essere private (non visibile da ext)
 - simile a static in function

Statinit.cpp

definizione

Quanti.cpp

type classname::varname = value

- -non viola la protezione private
- -è consentita solo una volta
- -è obbligatoria nella definizione della classe

AA 2001-2002

programmazione ad oggetti

7

static method in class

- function che opera su tutti gli oggetti istanziati dalla classe
- non ha riferimento ad un singolo oggetto (this)
- non può manipolare le variabili di classe ordinarie (solo quelle definite come static)
- non agisce su un singolo oggetto --> sintassi preferibile: scope resolution operator

class_name::static_function()

- solo per static method

StaticMemberFunction.cpp

copy constructor

- necessario tutte le volte che si vuole controllare come vengono create copie di oggetti esistenti
 - passaggio parametri per valore
 - creazione oggetti da esistenti
- default: copia byte per byte
 - non per i puntatori

AA 2001-2002

programmazione ad oggetti

^

passaggio parametri

- i parametri di una function vengono messi sullo stack
 - anche l'indirizzo del valore di ritorno
 - evita problemi di re-entrance (interrupt, ricorsione)
- passaggio per valore: valori sullo stack
- come fare per gli oggetti?
 - copia fisica della memoria
 - salta il costruttore !!!

CopyConst2.cpp

HowMany.cpp HowMany2.cpp