

Objektno orijentisano programiranje 1

Prostori imena

Problem konflikta imena

- Način povezivanja imena:
 - način na koji se povezuju ista imena (identifikatori) korišćeni u različitim datotekama
 - spoljašnje povezivanje (SP) imaju:
 - imena koja označavaju isti entitet (podatak ili funkciju) unutar svih datoteka programskog sistema
 - unutrašnje povezivanje (UP) imaju:
 - imena koja označavaju različite entitete u različitim datotekama, imena koja su lokalna za datoteku
- Na jezicima C i C++:
 - globalna imena imaju SP, osim ako se koristi `static` (tada je unutrašnje)
 - lokalna imena imaju UP, osim ako se koristi `extern` (tada je spoljašnje)
- Problem konflikta imena:
 - 2 globalna podatka sa SP, u 2 datoteke, dva programera mogu isto da imenuju
- Rešenje problema:
 - prostor imena (engl. *namespace*)

Definisanje prostora imena

- Prostor imena se definiše:
 - `namespace Identifikator {Sadržaj}`
 - *Identifikator* – ime prostora imena, jedinstveno u okružujućem dosegu
 - *Sadržaj* – programski tekst
- Unutar prostora imena, globalna imena uvedena u datom prostoru imena
 - mogu da se koriste neposredno
- Doseg globalnih imena je prostor imena unutar kojeg su uvedena
 - od mesta definicije ili deklaracije do kraja prostora
- Izvan prostora imena, korišćenje globalnih imena iz datog prostora imena
 - moguće je uz proširenje doseg, odnosno specijalne deklaracije korišćenja
- Prostor imena ne menja:
 - način povezivanja globalnih imena
 - životni vek globalnih podataka

Osobine prostora imena

- Osobina otvorenosti prostora imena:
 - više definicija sa istim identifikatorom obrazuje jedan prostor sa datim imenom
 - unija prostora imena
 - imena iz kasnijih definicija prostora imena ne mogu da se koriste pre tih definicija
- Jedinstven prostor imena može biti u više datoteka
 - sadržaj prostora imena se ne prenosi iz jedne u drugu datoteku
 - u datoteci može da se koristi samo navedeni deo sadržaja
 - potrebni delovi sadržaja moraju da se ponove u svakoj od datoteka
 - sadržaj prostora imena ne mora biti isti u svim datotekama
 - deklaracije funkcija i globalnih podataka treba pisati u .h fajlovima
 - da bi se pomoću `#include` direktive uključivale gde su potrebne
 - zato se i definicije klasa pišu u .h fajlovima (sadrže deklaracije metoda)

Primeri definisanja prostora imena

```
namespace A { // datoteka dat1.h
    int a = 100; // definicija podatka a
    double f(){return 0.0;} // definicija funkcije f()
    extern char b; // deklaracija podatka b
    float g(float); // deklaracija funkcije g(float)
}
namespace B {double a = 1.0;} // definicija druge promenljive a
namespace A {
    extern float x;
    char b = 'b'; // definicija podatka b
    char h(); // deklaracija procedure h()
}
#include "dat1.h" // datoteka dat1.cpp
namespace A {
    float x=0.0f; // definicija podatka x
    float g(float x) {return x/2;} // definicija funkcije g(float)
}
```

5

Prostori imena

01.12.2015.

Upotreba prostora imena

- Unutar prostora imena
 - sva globalna imena se dohvataju jednostavnim imenovanjem
 - ako su pre mesta dohvanja barem deklarirana
- Izvan prostora imena
 - imena iz prostora se mogu dohvatiti na jedan od načina:
 - (1) dohvanje pojedinačnog imena na mestu gde je potrebno
`Identifikator_prostora_imena::Ime_iz_prostora_imena`
 - (2) uvoz pojedinačnog imena
`using Identifikator_prostora_imena::Ime_iz_prostora_imena`
 - (3) uvoz svih imena iz prostora imena
`using namespace Identifikator_prostora_imena`

6

Prostori imena

01.12.2015.

Pravila upotrebe uvezenih imena

- Uvezena imena mogu da se koriste bez navođenja identifikatora prostora i operatora ::
 - pravilo važi samo ako su uvezena imena jednoznačna
 - ako postoji lokalno ime koje se preklapa sa uvezenim
 - samo ime predstavlja lokalno ime
 - imenu iz prostora imena se može pristupiti na način (1)
 - ako ne postoji lokalno ime, ali se imena iz više prostora preklapaju
 - načinom (2) može da se naznači prostor iz kojeg se koristi ime ako se navodi samo
- Globalni podaci i funkcije koji su deklarirani u prostoru imena mogu se definisati izvan tog prostora na način (1)
- Navođenjem identifikatora prostora imena ispred funkcije (uz ::) doseg prostora imena se proširuje na telo funkcije

7

Prostori imena

01.12.2015.

Primeri upotrebe prostora imena

```
int p = A::a;
double q = B::a;

using A::x;
float r = x;           //A::x

using namespace A;
a = 1;                //A::a
float s = g(x);       //A::g(A::x)

using namespace B;
int t = a;            // !GRESKA (viseznacno)
int u = A::a;
double v = B::a;

using A::a;
int w = a;            //A::a
```

8

Prostori imena

01.12.2015.

Anonimni (bezimeni) prostor imena

- Anonimni prostor imena čini “globalna” imena imenima sa unutrašnjim povezivanjem
 - to ni `extern` ne može da promeni
- Definiše se izostavljanjem identifikatora iz definicije prostora imena
- Anonimni prostor ne uvodi poseban doseg imena
- Imena iz anonimnog prostora
 - imaju datotečki doseg
 - moraju se razlikovati od običnih globalnih imena
 - mogu se koristiti navođenjem bez `::`
- Isto ime iz anonimnog prostora u dve datoteke označava različite entitete
- Anonimni prostor imena se preporučuje umesto stavljanja `static` za globalna imena
- Ako se uveze neki imenovani prostor imena
 - ista imena iz anonimnog prostora više ne mogu da se koriste

9

Prostori imena

01.12.2015.

Primer anonimnog prostora imena

```
namespace {
    int a=1;
}

int b=2;

namespace A{
    int a=3;
    int b=4;
}

void f(){
    int o=a;    //::a (iz anonimnog prostora)
    int p=b;    //::b (globalna promenljiva)
    int q=A::a;
    using namespace A;
    int r=a;    //GRESKA
    int s:::a;  //GRESKA (*)
    int t=A::a;
    int u=b;    //GRESKA: ::b ili A::b ?
    int v:::b;
    int w=A::b;
    int b=5;    // def. lokalne promenljive b
    int x=b;    // lokalno b
    int y:::b;  // globalno b
    int z=A::b;
}

// (*) ne moze se dohvatiti a
// iz anonimnog prostora imena
```

10

Prostori imena

01.12.2015.

Uklapanje prostora imena

- Moguće je definisanje prostora imena unutar definicije prostora imena
- Definicija uklopljenog prostora imena mora biti u "globalnom" delu spoljašnjeg prostora
- Identifikator uklopljenog prostora se koristi kao i sva imena u spoljašnjem prostoru (izvan spoljašnjeg prostora: `ime_spoljašnjeg::ime_unutrašnjeg`)
- Doseg imena spoljašnjeg prostora se proteže i na unutrašnji prostor
 - imena iz spoljašnjeg se neposredno koriste u unutrašnjem (ako su definisana ispred unutrašnjeg i ako nisu ponovo barem deklarirana u unutrašnjem)
 - ako postoji preklapanje imena iz spoljašnjeg sa imenom iz unutrašnjeg prostora, dohvatanje imena iz spoljašnjeg prostora:
`ime_spoljašnjeg_prostora::ime_iz_spoljašnjeg_prostora`
- Doseg unutrašnjeg prostora se ne proteže na spoljašnji prostor
 - imena iz unutrašnjeg se u spoljašnjem koriste na način
`ime_unutrašnjeg_prostora::ime_iz_unutrašnjeg_prostora`
 - to važi samo za deo spoljašnjeg prostora koji se nalazi iza definicije unutrašnjeg prostora

11

Prostori imena

01.12.2015.

Primer uklapanja prostora imena

```
namespace A{
    int a=1;
    int f() { return 2; }
    extern int b;
    int g();
    namespace B{
        int i;
        extern int a; //deklaracija B::a
        int f(); //deklaracija B::f()
        int g() { //definicija B::g()
            a++; //B::a
            return b; } //A::b
    }
    int c=a+B::a; //A::a+B::a
    int j=B::i; // ne moze samo i
}

void h(){
    int p=A::a;
    int q=A::f();
    int r=A::B::a;
    int s=A::B::f();

    using namespace A;
    int t=a; //A::a
    int u=f(); //A::f()
    int v=B::a; //A::B::a
    int w=B::f(); //A::B::f()
}
int A::b=3;
int A::g(){ return 4; }
int A::B::a=5;
int A::B::f(){ return b; } //A::b
```

12

Prostori imena

01.12.2015.

Standardni prostor imena

- Standard propisuje da standardna zaglavlja smeštaju imena u prostor `std`
- Standard predviđa da se standardna zaglavlja ne čuvaju u tekst-datotekama
 - zato imena standardnih datoteka-zaglavlja u jeziku C++ nemaju tip `.h`
- Na primer: `<iostream>` definiše sva imena u prostoru `std`, za razliku od `<iostream.h>` gde su imena globalna

- Primer:

```
#include <iostream>
void main(){
    int n;
    std::cout<<"n?";
    std::cin>>n;
    using namespace std;
    cout<<"n?";
    cin>>n;
}

#include <iostream.h>
void main(){
    int n;
    cout<<"n?";
    cin>>n;
}
```

13

Prostori imena

01.12.2015.

Zaglavlja standardnih biblioteka

- Uporedni pregled:

C++	C
<code><cstdlib></code>	<code><stdlib.h></code>
<code><cstdio></code>	<code><stdio.h></code>
<code><cmath></code>	<code><math.h></code>
<code><cctype></code>	<code><ctype.h></code>
<code><cstring></code>	<code><string.h></code>

- Standard ne preporučuje korišćenje `.h` zaglavlja u novim C++ programima
- Preporuka: navesti sva potrebna standardna zaglavlja naredbama `#include`, a zatim: `using namespace std`

- Primer:

```
#include <iostream>
#include <cmath>
using namespace std;
```

14

Prostori imena

01.12.2015.