Sveučilište u Zagrebu
PMF – Matematički odjel

Objektno programiranje (C++)

Predavanja 03 - Nasljeđivanje

Vinko Petričević

Nasljeđivanje

- Slični tipovi mogu imati neka zajednička svojstva i ponašanje

```
struct A {
    int x, y;
    A(int x, int y) : x(x), y(y) {}
};
void ispisi(A a){ cout << a.x << " " << a.y << endl; }

struct B : /*public*/ A {
    B(int x, int y) : A(x,y){}
};
...
    B b(1,2); ispisi(b);
```

- Sada radi a=b
- Obratno može biti opasno, pa ne radi. A ako to baš želimo, možemo naglasiti b=(B&)a;
- Public/private/protected
- Ako imamo dvije funkcije/varijable koje se jednako zovu, npr. print(). b.print() će pozvati verziju od B, a ako želimo pristupiti A funkciji, možemo pisati b.A::print() ili ((A&)b).print() (bez reference može biti opasno)

Polimorfizam

- Slični tipovi mogu imati različito ponašanje

```
    virtual void A::print() { cout << "A" << endl; }
void ispisi(A& a){ a.print(); }
    void B::print() { cout <<  "B" << endl; }

    ispisi(b);
```

- To radi i da na klasi A imamo nevirtualnu funkciju koja poziva print()

- Virtualni destruktor

- Kastanje u baznu klasu sa i bez &
- izvedenu klasu možemo od klase tipa A dobiti sa (B&), tj static_cast<B&>, no to može biti opasno
- Kod pointera ako je tip polimorfan (ima bar jednu virtualnu funkciju), radi i dynamic_cast<B*>, koji vraća null ako konverzija nije sigurna.

- Objasniti pointere na funkcije

Formatiranje streamova

- Primjeri s formatiranjem streamova
- Zadatak s vježbi gdje ispisujemo na različite streamove
- Budući je operator<< virtualan (ili je možda samo flush() virtualna) to će sve tako jednako raditi pišemo li u datoteku, string ili na ekran

- Razlike između endl i ‘\n’

- Dodavanje separatora

- Zadatak:
- Napraviti nešto slično endl-u na ofstreamu, samo da umjesto prelaska u novi red počne pisati u novu datoteku

Međuklasa za „poboljšanje” map[]

- Operator[] na mapi ubacuje element ukoliko element nije postojao u mapi
- Vraća referencu na value_type, pa nam to može ubrzati neke petlje

- Možemo napraviti da taj operator vraća neku međuklasu na kojoj onda napravimo operatore pridruživanja i castanja u value, pa možemo razlikovati pridruživanje nove vrijednosti od očitavanja stare.
- Tu međuklasu bi bilo dobro sakriti od javnosti da ne izaziva greške.

- Spomenuti const []

Višestruko i virtualno nasljeđivanje

- Za razliku od nekih drugih programskih jezika, C++ dopušta višestruko nasljeđivanje:
  class C : public A, public B { … };
- Klasa C je naslijedila sve i od klase A i klase B
- Ukoliko imamo elemente koji se jednako zovu, kada im pristupamo trebamo naglasiti sa A:: ili B::
- Ukoliko radimo castanje iz objekta tipa C u npr. B, automatski će se dodati sizeof(A), odnosno oduzeti ako radimo obnuto castanje (koje moramo naglasiti)
- Konstruktore dodajemo u inicijalizacijsku listu isto kao i kod običnog nasljeđivanja

- Ukoliko su klase A i B obje naslijeđene iz neke klase X, klasa C će imati dvije instance klase X
- Ponekad to i želimo
- Ako želimo imati jedinstveni X, kod nasljeđivanja iz X treba pisati virtual:
  class A : public virtual X {…}; class B : public virtual X {…}

- Konstruktori kod virtualnog – kod kreiranja klase C zanemaruju se pozivi konstruktora od X iz A i B, te se poziva konstruktor naveden u C
- Redoslijed izvršavanja konstruktora je drugačiji – prvo se kreiraju virtualni objekti, pa onda obični