Elektrotehnički fakultet u Beogradu
Katedra za računarsku tehniku i informatiku

Predmet: Objektno orijentisano programiranje
(OF2OO1, OS2OO1, OS3OOP, OE2OOP, OT2OOP)
Nastavnik: doc. dr Dragan Milićev
Ispitni rok: Januar 2007.
Datum: 12.2.2007.

Kandidat: _____

Broj Indeksa: _____ *E-mail:*_____

Ispit traje 3 sata. Dozvoljeno je korišćenje literature.

Pismeni ispit:		***Domaći zadaci:***	
Zadatak 1	_____/10	*Obavezni deo*	_____/40
Zadatak 2	_____/10	*Neobavezni deo*	_____/10
Zadatak 3	_____/10		
Zadatak 4	_____/10		
Zadatak 5	_____/20		
Ukupno na ispitu:	**_____/60**	***Ukupno na domaćem:***	**_____/50**

Ukupno: _____/110

Ocena: _____ (_____)

Napomena: Ukoliko u zadatku nešto nije dovoljno precizno definisano, student treba da uvede razumnu pretpostavku, da je uokviri (da bi se lakše prepoznala prilikom ocenjivanja) i da nastavi da izgrađuje preostali deo svog odgovora na temeljima uvedene pretpostavke. Ocenjivanje unutar potpitanja je po sistemu "sve ili ništa", odnosno nema parcijalnih poena. Kod pitanja koja imaju ponuđene odgovore treba **samo zaokružiti** jedan odgovor. Na ostala pitanja odgovarati **čitko, kratko i precizno**. Zadaci 1-4 su eleminatorni prema Pravilima predmeta.

1. (10 poena)

Dat je deo koda na jeziku C++. Za svaku liniju označenu brojevima 1-5 navesti da li je ispravna (prevodilac neće prijaviti grešku) ili nije (prevodilac će prijaviti grešku) – u srednju kolonu upisati „Da“ ili „Ne“. Ako linije nije ispravna, u trećoj koloni kratko navesti vrstu greške (uzrok).

```
void main () {
  int a[10];
  int* p=a; int* q;
  q  = &*p++;   // 1
  &q = *p++;    // 2
  *q = *p++;    // 3

  extern int f();
  void (*pf1)(int) = &f;  // 4
  int (*pf2)() = &f;      // 5
}
```

Odgovor:

Linija	Ispravna	Vrsta (uzrok) greške
1		
2		
3		
4		
5		

2. (10 poena)

Dat je sledeći program:

```
#include <iostream.h>

class Object {
public:
  Object (int i, Object* nxt=0) : myID(i), next(nxt) {}
  virtual ~Object () { if (next) delete next; }

  virtual void write();

protected:
  int myID;
  Object* next;
};

void Object::write () {
  if (next) next->write();
  cout<<", "<<myID;
}

class Comedian : public Object {
public:
  Comedian (int i, Object* nxt=0) : Object(i,nxt) {}
  virtual void write();
};

void Comedian::write () {
  cout<<myID;
}

void main () {
  Object* p = new Object(1,new Object(2,new Comedian(3,new Object(4,new
Comedian(5)))));
  p->write();
  delete p;
  p = new Object(3,new Object(2,new Object(1)));
  p->write();
  delete p;
}
```

Šta ispisuje ovaj program (odgovor ispisati tačno kako i program ispisuje)?

Odgovor:

3. (10 poena)

Na jeziku C++ realizovati klasu `Vector` koja apstrahuje vektor u Dekartovom koordinatom sistemu. Vektor se predstavlja tačkom `(x,y)` (vektor uvek polazi iz koordinatog početka), a koordinate su tipa `double`. Ova klasa treba da obezbedi sledeće:

- Inicijalizaciju vektora koordinatama, pri čemu je podrazumevana inicijalizacija na (0,0).
- Operacije poređenja na jednakost (`operator==`) i nejednakost (`operator!=`) dva vektora.
- Operacije sabiranja (`operator+`) i oduzimanja (`operator-`) dva vektora.
- Operaciju skalarnog proizvoda dva vektora (`operator*`).
- Operaciju množenja skalara i vektora (`operator*`).

4. (10 poena)

Potrebno je realizovati sledeći sistem klasa i njihovih relacija. *Dokument* (*Document*) može imati jednu ili više svojih *Verzija* (*DocumentVersion*). Prilikom kreiranja novog Dokumenta, kreira se njegova početna Verzija. Prilikom uništavanja Dokumenta, uništavaju se sve njegove Verzije. Svaka verzija ima svoju oznaku koja predstavlja redni broj te Verzije: kada se kreira nova Verzija Dokumenta, ovaj broj se uvećava za jedan.

(a) Nacrtati dijagram klasa koji prikazuje UML model opisanog sistema.

(b) Napisati kompletan C++ kod klase Dokument (*Document*) i svih njenih operacija, uključujući operaciju `createNewVersion()`.

5. (20 poena) Konstruktivni zadatak

Radi optimalne organizacije javnog skupa, potrebno je napraviti uprošćenu simulaciju potrošnje piva. Pri realizaciji sistema, poći od sledećih pretpostavki:

- Šanker je sve vreme koncerta zaposlen i konstantno toči pivo, koje raspoređuje na N mesta na šanku. Za točenje svakog piva potrebno je T (±10%) sekundi (ravnomerna raspodela). Mesta se opslužuju kružnim redosledom, počev od prvog.
- Na svakom od mesta se sigurno nalazi posetilac javnog skupa, koji konzumira pivo, koje se zadržava u organizmu posetioca Z (±30%) minuta, pre nego posetilac ode do sanitarnog čvora i izluči pivo. Ako posetilac u nekom trenutku u organizmu ima više od M piva, odlazi do sanitarnog čvora i izbacuje sva piva koja ima u organizmu.
- Zadržavanje posetioca u sanitarnom čvoru je isto za sve posetioce.

N i T, odnosno Z i M, ne moraju biti isti za sve šankere, odnosno posetioce. Potrebno je projektovati klase koje će podržati opisano ponašanje. Nije potrebno pisati programski kod koji stvara i povezuje opisane objekte. Pri ovome, može se koristiti i/ili menjati postojeći programski kod TSS.

a) (7 poena) Šanker.

b) (5 poena) Posetilac (osim izbacivanja svih piva).

c) (5 poena) Posetilac (izbacivanje svih piva).

d) (3 poena) Sanitarni čvor.