Elektrotehnički fakultet u Beogradu
Katedra za računarsku tehniku i informatiku

Predmet: Objektno orijentisano programiranje
(OF2OO1, OS2OO1, OS3OOP, OE2OOP, OE4OOP, OT2OOP)
Nastavnik: Prof. dr Dragan Milićev
Ispitni rok: Septembar 2008.
Datum: 01.09.2008.

Kandidat: _____

Broj Indeksa: _____ *E-mail:*_____

*Ispit ima dva dela ukupnog trajanja 3 sata. Na prvom delu **nije** dozvoljeno korišćenje literature. Na drugom delu **jeste** dozvoljeno korišćenje literature. Trajanje prvog dela student određuje prema ličnom nahođenju. Student dobija drugi deo ispita kad preda prvi deo.*

UPISATI SVOJE PODATKE I NA PRVOJ STRANI DRUGOG DELA ISPITA!

Pismeni ispit:		***Rad u toku semestra:***	
Zadatak 1	_____/10	*Projekat*	_____/30
Zadatak 2	_____/10	*Obavezni domaći zadaci*	_____/10
Zadatak 3	_____/10	*Neobavezni domaći zadaci*	_____/10
Zadatak 4	_____/10		
Zadatak 5	_____/20		
Ukupno na ispitu:	**_____/60**	***Ukupno u toku semestra:***	**_____/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 eliminatorni prema Pravilima predmeta.

1. (10 poena)

Date su sledeće deklaracije na jeziku C++:

```
int**  x1;           // 1
int&&  x2;           // 2
char*& x3;           // 3
char&* x4;           // 4
char*& x5=0;         // 5
char&* x6=0;         // 6
int*   x7[10];       // 7
int&   x8[10];       // 8
int**  x9[10];       // 9
int*** x10[1];       // 10
```

Za svaku od linija označenih sa 1-10 navesti da li je ispravna ili nije (već će prevodilac generisati grešku). U tabelu upisati samo „Da“ ili „Ne“.

Linija	Ispravna?
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	

2. (10 poena)

Šta ispisuje sledeći program? Prokomentarišite ovaj program u smislu izvršavanja i eventualnih problema i njihovog otklanjanja.

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

class Widget {
public:
  static  Widget* create(int);
  virtual ~Widget() { cout << "Widget\n"; }
  Widget() {}
};

class Gadget : public Widget {
protected:
    ~Gadget () { cout << "Gadget\n"; }
};

class Doodad : public Widget {
protected:
    ~Doodad () { cout << "Doodad\n"; }
};

Widget* Widget::create (int i) {
  if (i%2) return new Gadget();
  else return new Doodad();
}

void main () {
  Widget magicBox[10];
  for (int i=0; i<10; i++)
    magicBox[i] = *Widget::create(i);
}
```

Odgovor:

3. (10 poena)

U potpunosti implementirati klasu `OrderedCollection` koja predstavlja dvostruko ulančanu listu uređenih elemenata čiji su elementi tipa `Object*`. Ova klasa treba da obezbedi sledeće:

- Podrazumevanu inicijalizaciju (kao prazna kolekcija).
- Propisno uništavanje, uz dealokaciju zauzetog prostora (uključujući i sadržane objekte).
- Operaciju stavljanja elementa u kolekciju na odgovarajuće mesto, uz očuvanje uređenja.
- Operaciju uzimanja elementa sa početka i kraja liste (izbacuje i vraća prvi, odnosno poslednji element).
- Operacije koje vraćaju prvi i poslednji element liste, bez njihovog izbacivanja iz liste.

Klasa `Object` obezbeđuje sve operatorske funkcije za poređenje.

4. (10 poena)

Potrebno je realizovati sistem klasa i njihovih relacija, opisanih tekstom u nastavku. *Dokument* (*Document*) može da bude prosleđen na obradu nekom *Učesniku* (*Participant*) u poslovnoj kolaboraciji i komunikaciji, pri čemu Učesnik može biti *Zaposleni* (*Employee*) ili *Odsek* (*Department*). Odsek je organizaciona jedinica preduzeća čiji su članovi Zaposleni. Zaposleni može da pregleda Dokumente koji su mu upućeni na pregled ovakvim mehanizmom.

(a) Nacrtati dijagrame koji prikazuju UML model klasa opisanog sistema.

(b) Napisati C++ kod za klasu Učesnika, Odseka i Zaposlenog, prema navedenim zahtevima. Napisati i programski kod za sve operacije koje učestvuju u implementaciji operacije slanja Dokumenta Učesniku.

5. (20 poena)

Potrebno je realizovati uprošćenu simulaciju dešavanja u delu buregdžinice kome potrošači imaju pristup. Po ulasku u buregdžinicu, potrošač staje u red za čekanje u kome se zadržava najduže 5 minuta (ravnomerna raspodela), ako u redu ima drugih potrošača. U suprotnom, dolazi odmah do vitrine sa proizvodima. Sa druge strane vitrine je radnik koji uslužuje potrošače prema njihovoj želji. Potrošač će, po plaćanju, dobiti hranu spremnu za konzumaciju na nekom od 3 stola u buregdžinici. Potrošač bira sto za kojim u trenutku izbora sedi najmanje ljudi. Zadržavanje na pojedinom stolu je između 10 i 30 minuta (ravnomerna raspodela), nakon čega potrošač odlazi. Pretpostaviti da su kapacitet svakog stola i broj stolova uvek dovoljni da opsluže sve potrošače. Po završenoj simulaciji, potrebno je ispisati sledeće rezultate: zbirni promet, ukupan broj potrošača i prosečno zadržavanje potrošača za svaki od stolova.

Obezbediti da prilikom ispisa rezultata simulacije za svaki sto bude ispisano prosečno vreme zadržavanja potrošača, te da za svakog potrošača budu ispisana vreme zadržavanja u redu ispred vitrine i vreme provedeno za stolom. Pri rešavanju je dozvoljeno koristiti postojeći programski kod TSS uz eventualne potrebne izmene i dopune.

a) (10 poena) Klase **`Ulaz`**, **`Potrosac`** i **`Sto`**. U slučaju upotrebe TSS, navesti eventualne dopune/izmene u postojećim TSS klasama.

b) (5 poena) U skladu sa zahtevima zadatka, obezbediti **`Visitor`** klasu/klase sa metodama odgovarajućim klasama **`Ulaz`** i **`Sto`**. U slučaju upotrebe TSS, adekvatno upotrebiti postojeće **`Visitor`** klase.

c) (5 poena) Napisati programski kod koji stvara i ispravno povezuje odgovarajući broj primeraka opisanih klasa, te pokreće simulaciju i ispisuje tražene rezultate simulacije. Navesti klase koje su eventualno potrebne za kompletnost simulacije, ukratko ih opisati (samo imena i opis metoda, bez implementacije) i stvoriti odgovarajući broj njihovih primeraka. Na kraju, osloboditi korišćenu dinamičku memoriju.