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: Oktobar 2008.
Datum: 17.09.2008.

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)

Dat je deo koda na jeziku C++. Za svaku liniju označenu brojevima 1-10 navesti da li je ispravna (prevodilac neće prijaviti grešku) ili nije (prevodilac će prijaviti grešku) – u tabelu upisati „Da“ ili „Ne“.

```
class X {
protected:
  X (double=0.0);               // 1
private:
  X (int);                      // 2
};
```

Klasa `Y` definisana je na sledeći način,

```
class Y : public X {
public:
  *
};
```

s tim da na mestu linije označene sa * stoji jedna od sledećih linija:

```
  Y () {}                        // 3
  Y () : X(1) {}                 // 4
  Y (int) {}                     // 5
  Y (int i) : X(i) {}            // 6
  Y (int i) : X((double)i) {}    // 7
  Y (double) {}                  // 8
  Y (double d) : X(d) {}         // 9
  Y (double d) : X((int)d) {}    // 10
```

Odgovor:

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

2. (10 poena)

Dat je sledeći program na jeziku C++:

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

class X {
public:
  X (const char*);
  void  setChar(int index, char);
  const char* getStr () const { return s; }
private:
  char* s;
};

X::X (const char* str) {
  if (s = new char[strlen(str)+1]) strcpy(s,str);
}

void X::setChar (int i, char c) {
  if (0<=i && i<strlen(s)) s[i]=c;
}

void main () {
  X x1("X1");
  X x2(x1);

  cout<<x1.getStr()<<'\n';          _____________________________________

  cout<<x2.getStr()<<'\n';          _____________________________________

  x2.setChar(1,'3');

  cout<<x1.getStr()<<'\n';          _____________________________________

  cout<<x2.getStr()<<'\n';          _____________________________________

}
```

Pored svake linije koda sa ispisom na `cout`, na liniji sa desne strane ispisati šta ispisuje ta linija.

3. (10 poena)

Na jeziku C++ realizovati klasu `Clock` koja apstrahuje vremenski brojač koji se može pokrenuti da meri interval vremena. Ova klasa treba da obezbedi sledeće:

- Inicijalizaciju zadatim vremenskim intervalom koji treba da meri (interval je predstavljen kao ceo broj vremenskih jedinica).
- Pokretanje merenja vremena.
- Zaustavljanje merenja vremena.
- Kada istekne zadati interval vremena, brojač treba da pozove polimorfnu operaciju `timeout()` (indirektne) instance apstraktne klase `Timeout` na koju ukazuje pokazivač dostavljen kao opcioni argument konstruktora brojača.
- Operaciju koja vraća vreme koje je preostalo do kraja intervala koji se meri.

Protok jedne jedinice vremena signalizira se spolja pozivom operacije `tick()` ove klase.

4. (10 poena)

Potrebno je realizovati sistem klasa i njihovih relacija, opisanih tekstom u nastavku. *Potrošač* dolazi u buregdžinicu radi kupovine brze hrane, koju će upakovanu poneti sa sobom. Zavisno od ličnog ukusa potrošača i broja osoba za koje potrošač nabavlja hranu, željeni proizvodi će biti spakovani na određeni način. Sigurno je da će potrošač u trenutku izlaska iz buregdžinice nositi jednu *Kesu*. U toj kesi mogu se nalaziti spakovani proizvodi, ali i druge kese, koje mogu sadržati i proizvode i druge kese i tako redom. *Proizvod* može biti *Burek*, *Krompiruša*, *Zeljanica*, *Sirnica*, *Urmašica* ili *Baklava*. *Radnik* će spakovati kesu sa proizvodima prema porudžbini potrošača i naplatiti račun prema sadržaju kese.

(a) Nacrtati dijagrame koji prikazuju UML model klasa opisanog sistema. Naznačiti koji je DP uočljiv na ovom dijagramu.

(b) Napisati C++ kod za klase *Kesa*, *Proizvod* i *Krompiruša*.

(c) Napisati metodu *float Radnik::IznosRacuna()* koja, na osnovu sadržaja kese predate potrošaču, određuje cenu koju potrošač treba da plati.

5. (20 poena)

Potrebno je realizovati uprošćenu simulaciju dešavanja u šalter sali studentskog odseka prilikom upisa studenata u narednu školsku godinu. Pošto je potrebno obraditi veliki broj studenata u kratkom vremenskom roku, svaki od 5 šaltera obavlja istu funkciju prijema i obrade dokumenata studenata. Ispred svakog šaltera postoji red u kome je nekolicina studenata, u kome je prosečno vreme čekanja ravnomerno raspodeljeno između 2 i 4 minuta. Da bi bilo izbegnuto stvaranje gužve, na vratima šalter sale stoji redar koji studente koji dolaze iz hodnika usmerava na trenutno najmanje zauzeti šalter. Čim student obavi posao na šalteru, izlazi iz studentskog odseka bez zadržavanja i ometanja ostalih studenata.

Obezbediti da prilikom ispisa rezultata simulacije za svaki šalter bude ispisan broj opsluženih studenata, te da za svakog studenta bude ispisano vreme zadržavanja u šalter sali. Pri rešavanju je dozvoljeno koristiti postojeći programski kod TSS uz eventualne potrebne izmene i dopune.

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

b) (5 poena) Dodati u postojeće **`Visitor`** klase metode odgovarajuće opisanim klasama i postavci zadatka.

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.