Elektrotehnički fakultet u Beogradu
Katedra za računarsku tehniku i informatiku

Predmet: Programiranje u realnom vremenu
Nastavnik: Prof. dr Dragan Milićev
Asistent: Bojan Furlan
Ispitni rok: Januar 2012.
Datum: 17.1.2012.

Kandidat: _____

Broj Indeksa: _____ *E-mail:*_____

Ispit traje 3 sata. Dozvoljeno je korišćenje literature.

Zadatak 1	_____/10	*Zadatak 4*	_____/20
Zadatak 2	_____/10	*Zadatak 5*	_____/10
Zadatak 3	_____/20	*Projekat*	_____/40
Ukupno na ispitu:	**_____/70**	***Ukupno na projektu:***	**_____/40**

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**.

1. (10 poena) Tolerancija otkaza

U nekom sistemu visoke pouzdanosti ugrađeni su mehanizmi detekcije grešaka na osnovu strukturnih provera (provere integriteta struktura podataka). Posmatra se struktura stabla u kome je jedan čvor predstavljen objektom sledeće klase:

```
class TreeNode {
public:
  ...
private:
  Object* contents;
  TreeNode* parent;
  Collection<Object*> children;

  static void checkIntegrity (TreeNode* root) throws (TreeNode*);
};
```

Funkcija `checkIntegrity()` proverava sve što se može proveriti u smislu integriteta podataka stabla koje počinje od datog korena i podiže izuzetak koji ukazuje na čvor u kome je prvo otkrivena nekonzistentnost. Implementirati ovu funkciju. Pretpostaviti da šablonska klasa `Collection` obezbeđuje odgovarajući iterator.

Rešenje:

2. (10 poena) Konkurentno programiranje

Fibonačijevi brojevi predstavljaju sekvencu celobrojnih vrednosti opisanih sledećom formulom:

$F_0 = 0$

$F_1 = 1$

$F_n = F_{n-2} + F_{n-1}$, za $n > 1$

Primer rezultujuće sekvenca je: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, . . .

Koristeći mehanizam randevua na jeziku Ada implementirati proces koji za zadati stepen *n* rekurzivno računa Fibonačijev broj *F(n)*, tako što kreira 2 podprocesa koji računaju *F(n-1)* i *F(n-2)*. Na kraju izračunavanja je potrebno ispisati dobijenu vrednost pomoću metode `Ada.Text_IO.Put_Line`.

Rešenje:

3. (20 poena) Realno vreme

U nekom sistemu za nadzor nekog osetljivog hemijskog procesa jedna kritična fizička veličina prati se senzorom koji je na računar povezan preko A/D konvertora čije je maksimalno vreme konverzije t_{ad} = 40 ms. Konvertor može da izmeri ovu veličinu samo ako je ona u određenom opsegu. Zbog toga senzor postavlja vrednost jednog svog registra kada ova veličina uđe u dati merni opseg, što predstavlja diskretan događaj u sistemu. Od tog trenutka, potrebno je u roku od t_c = 200 ms očitati praćenu veličinu i njenu vrednost ispisati na izlaz. Program za nadzor i upravljanje ovog sistema implementiran je kao jedan periodični proces koji je stalno aktivan.

Na raspolaganju su sledeće funkcije interfejsa prema uređajima:

`isMeasurable():Boolean`	Vraća True ako je vrednost u mernom opsegu
`startAD()`	Pokreće A/D konverziju
`readAD():Real`	Očitava A/D konvertor.

(a)(10) Na jeziku Ada implementirati ovaj proces tako da ne pati od kumulativnog plivanja (engl. *cumulative drift*) i ima zaštitu od prekoračenja vremenskog roka, uzimajući vremenske parametre kao simboličke konstante.

(b)(10) Napisati i objasniti nejednakosti koje predstavljaju uslove za periodu (T) i vremenski rok (D) ovog procesa u funkciji t_{ad} i t_c, a potom odrediti ove parametre (T i D).

Rešenje:

4. (20 poena) Raspoređivanje i rasporedivost

Data su četiri procesa sledećih karakteristika:

Proces	*Trenutak aktivacije*	*D*	*C*	*Prioritet za FPS*
a	4	10	4	4 (najviši)
b	2	14	6	3
c	2	8	3	2
d	0	8	4	1

Vrednost za vremenski rok *D* dat je relativno u odnosu na trenutak aktivacije.
Nacrtati vremenski dijagram raspoređivanja ovih procesa za interval od 0 do 18 ako je algoritam raspoređivanja:

a)(10) FPS (*Fixed Priority Scheduling*), sa raspodelom prioriteta koja je data u poslednjoj koloni gornje tabele;

b)(10) EDF (*Earliest Deadline First*).

Rešenje:

5. (10 poena) ROOM

Na jeziku ROOM dati rešenje problema filozofa koji večeraju (*dining philosophers*) koje nema problem izgladnjivanja (*starvation*), živog (*livelock*), ni mrtvog blokiranja (*deadlock*). Filozofe predstaviti klasom aktera, a sinhronizaciju obezbediti pomoću jednog centralnog aktera u vidu konobara (*waiter*) koji komunicira sa filozofima. Na strukturnom dijagramu (*structural diagram*) prikazati rešenje sa 3 filozofa.

Rešenje: