Elektrotehnički fakultet u Beogradu
Katedra za računarsku tehniku i informatiku

Predmet: Objektno orijentisano programiranje
(OE2OOP, OE4OOP, OF2OO1, OS2OO1, OS3OOP,
OT2OOP, OT3OOP)
Nastavnik: Prof. dr Dragan Milićev
Ispitni rok: Septembar 2014.
Datum: 20.08.2014.

Kandidat: _____

Broj Indeksa: _____ *E-mail:*_____

Ispit traje 3 sata. Prvih sat vremena nije dozvoljeno napuštati ispit.

Nije dozvoljeno korišćenje literature.

Pismeni ispit:		***Rad u toku semestra:***	
Zadatak 1	_____/10	*Projekat*	_____/20
Zadatak 2	_____/10	*Domaći zadaci*	_____/30
Zadatak 3	_____/10		
Zadatak 4	_____/20		
Ukupno na ispitu:	**_____/50**	***Ukupno u toku semestra:***	**_____/50**

Ukupno: _____/100

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)

Odgovoriti kratko i precizno na sledeća pitanja.

A) Da li se prevodi priloženi program? Ako se ne prevodi, šta je potrebno dodati da se prevodi? Obrazložiti odgovor.

```
#include <iostream>
using namespace std;
class X {
  int val;
public:
  X(int v) {val = v;}
  inline int f(X x);
};
int f(X x)
{ return x.val; }

void main() { f(X(2)); }
```

B) Da li se prevodi sledeći program napisan na jeziku C++? Obrazložiti odgovor.

```
#include <iostream>
using namespace std;
class Y{
protected:
  int val;
public:
  Y(int d) { val = d; }
};
class X {
private:
  Y y;
public:
  X(int v) { y.val = v; }
};
void main() { X x(3); }
```

C) Dovršiti definiciju klase X tako da važe uslovi navedeni u sledećem listingu programa:

```
X x1 = 1;                 // PREVODI SE.
X x2 = x1;                // PREVODI SE.
cout << -x1+x2 << endl;   // PREVODI SE I ISPISUJE "X(0)".
```

```
#include <iostream>
using namespace std;
class X {
  int *val;
  // ...
};
```

2. (10 poena)

Dat je sledeći program na jeziku C++:

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

class B {
      int v;
public:
      B (int i=0): v(i) { cout << 1 << v; }
      B (const B& b): v(b.v) { cout << 2 << v; }
      virtual ~B() { cout << 9 << v; }
      B& operator= (const B& b) { v=b.v; cout<<"x"; return *this; }
      virtual bool check () { return v > 0; }
};

class D : public B {
      B myB;
public:
      D (int v=6): B(v) { cout << 3; }
      virtual ~D() { cout << 8; }
      virtual bool check () { return (myB=3).check(); }
};

void main() {
      D d = 6;
      d.check ();
}
```

Šta ispisuje dati program? Detaljno obrazložiti odgovor.

Odgovor:

3. (10 poena)

Realizovati C++ klasu koja X koja mora da zadovolji sledeće kriterijume.

- Sadrži sopstveni dinamički alociran niz znakova, koji ne sme biti deljen između više drugih objekata date klase. Stvara se zadavanjem nepraznog niza znakova. Ukoliko se prosledi prazan niz znakova prijavljuje se greška izuzetkom IllegalArgumentException.
- Svaki objekat prilikom pravljenja dobija jedinstveni celobrojni identifikacioni broj, koji može da se pročita.
- Omogućava dodeljivanje jednog objekta klase drugom.
- Omogućava pravljenje objekta kao kopije već postojećeg objekta.
- Preklopljenim operatorom `()` dohvata se broj pojavljivanja zadatog znaka u nizu znakova.
- Ima preklopljen operator `==` kojim se proverava, da li su dva objekta jednaka po sadržaju.
- Omogućava ispis stanja objekta na izlaz preklopljenim operatorom `<<` u formatu `(ID, "niz znakova")`.
- Omogućava propisno uništavanje objekata.

4. (20 poena)

U nekom sistemu svi objekti apstraktnog tipa `View`, zainteresovani za prikaz nekih izvornih podataka, prijavljuju se jedinom objektu *Singleton* klase `Controller`. Objekat `Controller` vodi evidenciju o proizvoljno mnogo izvora podataka koji su apstrahovani tipom `Model`. Kada se neki objekat tipa `View` napravi, potrebno je da se prijavi objektu `Controller`, kako bi ga ovaj uključio u svoju evidenciju. Kada se neki objekat tipa `View` uništava, potrebno je da se odjavi objektu `Controller`, kako bi ga ovaj isključio iz svoje evidencije. Kada se informacija koju čuva neki `Model` promeni, taj objekat obaveštava `Controller` o promeni, dostavljajući mu pokazivač na sebe. `Controller` tada obaveštava sve objekte tipa `View` u svojoj evidenciji o toj promeni, pozivom njihove polimorfne operacije `update()`. Opisani projektni obrasac se u literaturi naziva Posmatrač (*Observer*).

a) Prikazati UML dijagram klasa koji modeluje navedeni sistem.

b) Napisati kompletan kod za klasu `Controller`.