Zavod za elektroniku, mikroelektroniku
računalne i inteligentne sustave

11. lipnja 2012.

Oblikovni obrasci u programiranju

završni ispit

Napomena uz sve zadatke: potrebno je prikazati vezu između komponenata predloženog rješenja i sudionika oblikovnog obrasca, te skicirati implementaciju u C++-u, C#-u, Javi ili Pythonu.

1. Razmatramo program za vektorsku grafiku koji treba iscrtavati geometrijske likove poput trokuta, kvadrata, pravocrtnih segmenata i tekstovnih oznaka. Iscrtavanje se treba moći transparentno provesti pod Microsoftovim Windowsima i XWindowsima, kao i u datotečni format SVG. Predloži organizaciju koja bi omogućila istovremenu mogućnost dodavanja novih geometrijskih likova te novih načina iscrtavanja. Pripazi da u rješenju minimiziraš ponavljanje.

2. Razmatramo oblikovanje programa za internetsku prodaju elektroničkih komponenata. Program mora omogućiti kupovinu jednostavnih artikala poput matične ploče, kao i složenih artikala koji se sastoje od većeg broja jednostavnih artikala. Složeni artikli mogu sadržavati druge složene artikle: npr. stolna konfiguracija sastoji se od računala, monitora i zvučnika, a računalo se opet sastoji od matične ploče, kućišta itd. Radi jednostavnosti, sve artikle koje je korisnik izabrao tijekom jedne kupovine također treba predstaviti složenim artiklom. Predloži rješenje koje bi omogućilo predstavljanje složenih artikala, računanje njihove ukupne cijene, kao i jednostavno stvaranje složenog artikla iz postojećeg (npr. kad kupcima želimo ponuditi jednostavan odabir unaprijed konfiguriranih složenih artikala).

3. Zadana je funkcija za izračunavanje Fibonaccijevih brojeva, te ispitni program.

```
def fib(n):
  if n < 2:
    return n
  return fib(n-1) + fib(n-2)

def test():
  print([fib(n) for n in range(5)])
```

   Koliko puta će se pozvati funkcija `fib` pri evaluiranju ispitnog programa? Kako bismo taj broj poziva mogli smanjiti primjenom cacheirajućeg Proxyja?

4. Razmatramo oblikovanje biblioteke za olakšavanje razvoja jednostavnih prilagođenih HTTP poslužitelja. Biblioteka treba pružiti funkcionalnost za prihvaćanje spajanja mrežnih klijenata, te omogućiti klijentima da konkretnu obradu zahtjeva GET i POST definiraju vlastitim kôdom. Pretpostavljamo da sve zahtjeve treba obraditi isti kôd neovisno o zatraženoj stazi. Predloži organizaciju koja bi klijentima omogućila postizanje opisane funkcionalnosti nasljeđivanjem, i uz što manju količinu vlastitog kôda.

5. Korisnik izrađuje aplikaciju s grafičkim korisničkim sučeljem. U okviru te aplikacije potrebno je podržati sustav izbornika. Korisnik međutim za to želi koristiti ugrađenu biblioteku koja treba omogućiti lagano dodavanje izborničkih stavki i kôda koji te stavke trebaju pokrenuti. Predložite organizaciju takve biblioteke. Koji je oblikovni obrazac prikladan? U vašem rješenju nacrtajte dijagram razreda i naznačite sudionike tog oblikovnog obrasca. Prikažite to na primjeru stavki: "Kopiraj", "Izreži" i "Zalijepi".

6. Potrebno je uporabom oblikovnog obrasca Iterator omogućiti "posjetu" elemenata skupnog objekta ParniBrojevi koji predstavlja sortiranu kolekciju uzastopnih parnih brojeva. Neka konstruktor tog razreda prima prvi parni broj te broj uzastopnih parnih brojeva koji pripadaju toj kolekciji. Niti skupni objekt niti iterator ne smiju ni u jednom trenutku doista stvoriti u memoriji čitavu takvu kolekciju. Primjer uporabe u jeziku Java prikazan je u nastavku.

```
ParniBrojevi pb = new ParniBrojevi(2, 100); // <== ne sprema kolekciju
Iterator<Integer> it = pb.iterator();
```

Predložite cjelokupnu programsku implementaciju svih potrebnih razreda. Iterator treba biti izveden kao korisnički razred. Ispitni program mora uporabom primjerka iteratora ispisati sve elemente skupnog objekta.