

R smer, Dizajn programskih jezika, primer ispita

NAPOMENA: 1. i 2. zadatak se rade u programskom jeziku GO, a 3. i 4. u programskom jeziku C++. Potrebno je da se strogo držite formata ispisa koji je naznačen u zadacima.

1. (6 p) Napisati konkurentan program koji nasumično generiše n pozitivnih dvocifrenih brojeva i određuje proizvod cifara desetica svih brojeva. Koristiti zasebne gorutine za obradu određenih brojeva. Broj gorutina treba da odgovara broju procesora koji su na raspolaganju. Nakon obrade svih brojeva, glavna (**main**) gorutina treba samo da prikaže izračunat proizvod cifara. Pretpostaviti da su ulazni podaci ispravni.

UPUTSTVO: Pozitivni dvocifreni brojevi se mogu dobiti od random generisanog broja iz intervala $[0, 90)$ kome je dodat broj 10.

NAPOMENA: Zadatak se može rešiti sinhronizacijom niti preko katanca ili bez sinhronizacije, podelom poslova po nitima unapred

2. (9 p) Zimsko vreme je idealno za spremanje poslastica. Pet prijateljica je odlučilo da organizuje okupljanje i tom prilikom će napraviti različite poslastice. Međutim, svakoj od njih nedostaje nešto od sledećih sastojaka: jagode, sladoled, čokolada ili banane. Postoje četiri prodavnice u kojima mogu nabaviti te sastojke. U svakoj mogu nabaviti tačno jedan sastojak. Napisati program koji simulira kupovinu ovih pet prijateljica. One kreću iz nasumično odabranih prodavnica (dozvoljeno je da više njih krene iz iste prodavnice), idu redom i kupuju potrebne sastojke, a kada sve završe kupovinu program se zaustavlja.

Prodavnice: Prodavnice dopunjuju svoje zalihe do maksimalnog kapaciteta periodično na svakih 5 sekundi. Prodavnice su deljeni resursi, iz razloga što u datom momentu više kupaca može čekati na robu. Mehanizam čekanja u prodavnicama realizovati korišćenjem uslovnog reda čekanja. Svaka prodavnica od informacija ima jedinstveni identifikacioni broj, trenutno stanje, maksimalni kapacitet i oznaku J, S, C ili B u zavisnosti koji sastojak se prodaje u toj prodavnici.

Prijateljice: U slučaju da prodavnica trenutno nema potrebnu količinu robe, prijateljica čeka dok se zaliha ne obnovi. Kad završi kupovinu u jednoj prodavnici, kreće u sledeću nasumično odabranu prodavnicu. Potrebno je kreirati pet prijateljica i inicijalno ih postaviti na nasumično odabrane polazne prodavnice. Takođe, inicijalno, korpe svih njih su prazne. Svaku prijateljicu treba implementirati kao zasebnu gorutinu. Pri kreiranju prijateljice proslediti potrebne količine jagoda, sladoleda, čokolada ili banana (random izabrani brojevi od 0 do 1000 koji se odnose na gramažu odgovarajućeg proizvoda).

Modelovati opisan sistem kupovine i prikazivati informacije kad god neka od prijateljica uđe u prodavnicu, kad se obavi kupovina i finalno, kad je završeno sa nabavkom svih sastojaka.

3. (5 p) Napisati šablon klasu **PravaKupa** u okviru prostora imena **obrtnaTela** koja kao polja podatke ima dužinu poluprečnika osnove r i dužinu visine H , oba tipa T koji je parametar šablona i u njoj obezbediti:

- (a) konstruktor sa dva argumenta
- (b) metod za računanje zapremine prave kupe (koristiti konstantu **M_PI** definisanu u zaglavlju **cmath**), pretpostaviti da za objekte tipa T postoji definisano množenje sa konstantama i da će rezultat izračunavanja biti tipa T
- (c) operatore poređenja $<$, $=$ (poređenje dve kupe se svodi na poređenje njihovih zapremina, pretpostaviti da za objekte tipa T postoje definisani operatori poređenja)

Napisati program koji testira klasu **PravaKupa<double>**. Program prvo učitava podatke za dve kupe sa realnim poljima i ukoliko su jednake ispisuje apsolutnu vrednost razlike njihovih zapremina, inače zapreminu bilo koje kupe. U slučaju neispravnih ulaznih podataka ispisati -1 i prekinuti program.

Primer 1*

```
|| ULAZ:
|| 5
|| 4.5
|| 4.5
|| 6.5
|| IZLAZ:
|| 20.0277
```

Primer 2

```
|| ULAZ:
|| 6
|| 2
|| 3
|| 8
|| IZLAZ:
|| 75.3982
```

Primer 3

```
|| ULAZ:
|| 4.4
|| -5.4
|| IZLAZ:
|| -1
```

Primer 4

```
|| ULAZ:
|| 4.4
|| 5.4
|| 10.1
|| -52.4
|| IZLAZ:
|| -1
```

4. (10 p) Nemanja želi da obraduje svoju sestru za rođendan i kupi joj poklon. On na raspolaganju ima određenu količinu novca i zanima ga koji to artikal može da priušti. Napisati program koji pomaže Nemanji u kupovini poklona na osnovu podataka o artiklima (naziva i njihovih cena) pri čemu je kriterijum izbora količina novca kojom raspolaže.
- a) Napisati funkcional pod nazivom *Povoljno* koji za argument prihvata vrednost koja predstavlja količinu novca kojom kupac raspolaže i služi za proveru da li neki artikal može biti kupljen tim novcem ili ne. Ukoliko vrednost prosleđena konstruktoru nije ispravna, podrazumevano postaviti količinu novca na 2000.
- b) Napisati šablon funkciju koja za argumente dobija kolekciju sa podacima o artiklima i proizvoljan funkcional i ispisuje nazive artikala koji mogu da se priušte za datu količinu novca ili odgovarajuću poruku ukoliko takvih nema.
- c) Sa standardnog ulaza se unosi ukupan broj artikala (ceo pozitivan broj), a zatim podaci o artiklima i količina novca kojom Nemanja raspolaže (ceo pozitivan broj). Podaci o artiklima sadrže *Naziv* (pretpostaviti da ce biti jedinstvena niska karaktera) i *Cenu* svakog artikla (realan broj). Pretpostaviti da su svi podaci ispravno zadati. Na standardni izlaz ispisati:
- 1) učitane podatke o artiklima u obliku *Naziv Cena* sortirano opadajuće po ulaznom podatku *Naziv*.
 - 2) nazive artikala koje Nemanja može da kupi (sortirano opadajuće po nazivu).

Primer 1*

```

|| ULAZ:
|| 3
|| mindjuse 999.00
|| ogrlica 1950.00
|| narukvica 1500.00
|| 1000
|| IZLAZ:
|| ogrlica 1950.00
|| narukvica 1500.00
|| mindjuse 999.00
|| mindjuse

```

Primer 2

```

|| ULAZ:
|| 3
|| parfem 3780.00
|| sesir 900.00
|| marama 1000.00
|| 3000
|| IZLAZ:
|| sesir 900.00
|| parfem 3780.00
|| marama 1000.00
|| sesir
|| marama

```

Primer 3

```

|| ULAZ:
|| 2
|| sesir 1500.00
|| ogrlica 2500.00
|| 900
|| IZLAZ:
|| sesir 1500.00
|| ogrlica 2500.00
|| Nema povoljnih artikala.

```

Primer 4

```

|| ULAZ:
|| 2
|| ogrlica 2500.00
|| sesir 500.00
|| -100
|| IZLAZ:
|| sesir 500.00
|| ogrlica 2500.00
|| sesir

```