PRAKTIKUM IZ PROGRAMIRANJA 2

- domaći zadatak broj 4 -

Sastaviti program na programskom jeziku C koji vrši određenu vrstu obrade nad jednom ili više jednostruko ulančanih lista. Program treba da omogući učitavanje svih potrebnih ulaznih podataka, njihovo ispisivanje, obradu, ispis dobijenih rezultata, i ponavljanje navedenih koraka sve dok korisnik ne unese vrednost koja označava kraj programa. **Navedene korake u izvršavanju programa realizovati kao zasebne funkcije.**

Zavisno od rednog broja problema, sastaviti **jedan** od sledećih programa, koji:

0. simulira kretanje lifta; ulančana lista sadrži podatke o osobama: sprat na kome se osoba nalazi, i sprat na koji želi da stigne, kao i masu i ime osobe; lift se karakteriše maksimalnom dozvoljenom masom i rednim brojevima spratova između kojih se kreće; može postojati proizvoljan broj osoba prisutnih u liftu; sa standardnog ulaza se vrši učitavanje maksimalne mase za lift i polaznog i odredišnog sprata između kojih se kreće lift, kao i podataka o osobama koje čekaju lift; zatim se simulira kretanje lifta: lift se zaustavlja na svakom spratu i određeni broj osoba može izaći ukoliko su stigle do svog odredišta, pri čemu se masa koju lift prenosi ažurira, a potom se u listi pronađu podaci o osobama koje na tom spratu čekaju lift; ukoliko masa osobe dodata na trenutnu masu koju lift prenosi ne premašuje maksimalnu dozvoljenu masu i ukoliko ta osoba ide na sprat koji se nalazi u smeru kretanja lifta, ona može ući u lift; nakon toga, ispisuju se podaci o osobama prisutnim u liftu; pre ponovnih polazaka lifta sa polaznog sprata, zahtevati od korisnika da unese da li želi da nastavi simulaciju;
1. obrađuje ulančane liste koje sadrže termine časova na fakultetu; podaci o terminu se sastoje od početnog i krajnjeg sata (od kada-do kada) predavanja i rednog broja dana u nedelji kada se predavanje održava, pri čemu termini u listi ne moraju biti sortirani po rednom broju dana; obrada se sastoji od učitavanja termina u kojima nastavnik želi da drži časove u jednu listu i učitavanja mogućih termina za časove u drugu listu; nakon učitavanja svih podataka, treba napraviti raspored časova, tako da se poštuju predlozi termina koje je dao nastavnik: ukoliko je termin koji je nastavnik predložio moguć (proveru vršiti na osnovu raspoloživosti termina u drugoj listi) prihvata se; za željene termine za koje ne postoje odgovarajući slobodni termini, bira se neki od preostalih slobodnih termina koji su u istom danu kada i željeni termin; ukoliko ni takav ne postoji, odabrati bilo koji od preostalih slobodnih termina, ukoliko postoje; raspored časova se pravi tako što se termini dodaju u novu listu; nakon obrade, ispisati napravljeni raspored časova, kao i željene termine koji nisu raspoređeni;
2. simulira dodeljivanje sedišta i ukrcavanje putnika u avion na letu neke američke aviokompanije; američke kompanije sprovode praksu „overbooking"-a kako bi obezbedile 100% popunjenost kapaciteta aviona; aviokompanija poseduje spisak ljudi koji putuju na zadatom letu; podatke o jednom putniku čine ime, klasa u kojoj putuje (biznis ili ekonomska) i specijalna oznaka prioriteta (R – regularan putnik, I – osoba sa posebnim potrebama, P – trudnice, F – osobe sa malom decom, S – osobe starije od 65 godina); ukrcavanje se vrši u nekoliko tura; najpre se ukrcavaju putnici u biznis klasi, a zatim regularni putnici, vodeći računa da u svakoj klasi prioritet imaju putnici koji nisu regularni i oni najpre treba da se ukrcaju i dobiju sedište u avionu; broj sedišta u avionu po klasama se učitava sa standardnog ulaza; potrebno je svim putnicima dodeliti broj sedišta redosledom kojim će biti prozivani radi ulaska u avion; broj sedišta se sastoji se od dva znaka: reda obeleženog brojem i sedišta obeleženog slovom abecede od A do F; putnicima kojima nije dodeljeno sedište usled „overbooking"-a, dodeliti XX; putnici se mogu premeštati po klasama ukoliko nema raspoloživih sedišta u njihovoj klasi;

3. simulira razmenu sličica između dve osobe; svaka osoba poseduje dva spiska sličica; jedan je spisak sličica koje osoba poseduje za razmenu (duplikati), a drugi je spisak sličica koje joj nedostaju; pretpostaviti da jedna osoba poseduje samo jedan duplikat; spisak sličica se predstavlja u vidu ulančane liste koja sadrži redni broj sličice u albumu i ime i prezime igrača koji se nalazi na sličici; na osnovu zadatih spiskova svake osobe, potrebno je formirati spisak sličica koje nedostaju prvoj osobi, a poseduje druga i obrnuto; na kraju, omogućiti razmenu **n** sličica, pri čemu se **n** unosi sa standardnog ulaza;
4. simulira igru „aračkinje baračkinje"; u igri učestvuju dva tima igrača poređanih u dve vrste (ulančane liste) pri čemu se igrači jednog tima drže za ruke; za svakog igrača se unosi ime i snaga u napadu i odbrani (realan broj); jedan potez igre se sastoji iz sledećeg: igrač tima koji napada se zaleće iz svoje vrste i trči ka suparničkoj pokušavajući da probije lanac; ukoliko je jačina njegove snage napada veća od zbira jačina snage odbrane igrača koje on hoće da rastavi, on uspeva da ih rastavi i može da izabere igrača suparničkog tima koga će kao pobednik tog poteza dovesti nazad u svoj tim; ukoliko mu je napad slabiji od odbrane on ostaje u suparničkom timu na mestu gde je hteo da probije lanac; za svaki napad se putem standardnog ulaza unosi redni broj igrača koji napada i redni broj levog igrača od para igrača koje će pokušati da rastavi, kao i redni broj igrača koga bi doveo u svoj tim ako bi bio jači; potrebno je ažurirati i prikazati stanje lista nakon svakog poteza; nakon prvog tima na potezu je uvek drugi; igra se završava kada u nekom timu ostane jedan igrač;
5. predstavlja simulaciju rešavanja „Josephus problem"-a; određeni broj osoba (predstavljenih svojim imenima i prezimenima u vidu ulančane liste) je raspoređeno u krug i svaki korak rešavanja problema se sastoji od izbacivanja jedne osobe iz kruga, a zatim preskakanja **n** narednih osoba; pozicija u krugu osobe koja se izbacuje na samom početku i broj **n** se unose sa standardnog ulaza; potrebno je prikazati stanje liste osoba nakon svakog kruga sve dok u listi ne preostane jedna osoba;

Sve funkcije smestiti u odgovarajuće **.c** datoteke (prema donjem spisku), a prototipove svih funkcija smestiti u zajedničku **.h** datoteku. Učitavanje liste/lista realizovati pomoću funkcije kojoj će kao argument biti dostavljena adresa pokazivača na početak liste i broj elemenata koji treba učitati, a koja preko povratne vrednosti vraća podatak o uspešnosti učitavanja. Funkcijama koje vrše obradu treba dostaviti samo neophodne podatke (pokazivač na početak liste i, po potrebi, podatke od kojih zavisi obrada). Potrebno je napisati funkcije koje vrše ispisivanje liste, brisanje liste, ubacivanje elementa na početak i na kraj liste i izbacivanje proizvoljnog elementa iz liste.

Napraviti interaktivni meni kojim se omogućava učitavanje liste, brisanje liste, izbacivanje elementa iz liste, ubacivanje elemenata na početak i kraj liste, ispisivanje liste, obrada liste i prekidanje programa. **Ukoliko program radi sa više listi (varijante 1, 3, 4) obezbediti posebne stavke menija (koje su potrebne) za svaku listu. Ukoliko se obrada sastoji iz više koraka (varijante 4 i 5) obezbediti posebne stavke menija za svaki korak.** Voditi računa o pravilnom alociranju i dealociranju dinamičke memorije. **Potprogrami ne smeju komunicirati pomoću globalnih promenljivih, već samo preko liste argumenata i povratne vrednosti.**

Ako nešto u postavci zadatka nije dovoljno precizno definisano ili ako su neki od zahteva međusobno suprotstavljeni, usvojiti razumnu pretpostavku i rešiti zadatak korišćenjem te pretpostavke. Odabrati nekoliko skupova podataka sa kojima će program biti testiran. Odabrane test primere priložiti na listu papira pre odbrane. Kandidati koji na odbrani nemaju spremna makar tri suštinski različita test primera ne mogu dobiti maksimalan broj poena. Programski kod rešenja zadatka treba da bude uredno komentarisan, tako da pri pregledu programa lako može biti uočeno šta radi bilo koja programska celina. Takođe, treba poštovati pravila nazubljivanja (identacije) određenih celina prilikom pisanja koda.

Napomene:

1. Odbrana četvrtog domaćeg zadatka je u nedelji od 21.05. do 23.05.2014. po rasporedu dostupnom na sajtu predmeta.
2. Formula za redni broj problema **i** koji treba rešavati je sledeća (R – redni broj indeksa, G – poslednje dve cifre godine upisa): **i = (R + G) mod 6**
3. Kao rešenje domaćeg zadatka potrebno je na odbrani pokazati sledeće datoteke:
   - **`dz4.c`**, koja sadrži izvorni tekst osnovnog programa na programskom jeziku C;
   - **`dz4.h`**, koja sadrži prototipove svih funkcija opisanih u postavci zadatka;
   - **`dz4_unos.c`**, **`dz4_ispis.c`**, **`dz4_obrada.c`**, koje sadrže izvorne tekstove potrebnih funkcija.

13.05.2014. godine *sa predmeta*