PRAKTIKUM IZ PROGRAMIRANJA 2

- domaći zadatak broj 4 -

Sastaviti program na programskom jeziku C kojim se vrši određena vrsta 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, kojim se:

0. iz date liste elemenata koji predstavljaju kružnice, zadate koordinatama centra x i y i poluprečnikom r, izbacuju sve one koje se presecaju više od n puta sa drugim kružnicama u listi; izbačene elemente smestiti u novu listu;
1. izračunava zbir dva pozitivna cela broja predstavljena sa dve liste čiji elementi sadrže cifre u heksadecimalnom brojnom sistemu; rezultat smestiti u istu takvu listu; iz svake od lista izbaciti sve heksadecimalne cifre i ponoviti sabiranje, ovaj put u decimalnom brojnom sistemu;
2. iz liste tačaka u ravni izbacuju one tačke koje su u svom kvadrantu najudaljenije od koordinatnog početka; izbačene tačke smestiti u novu listu;
3. iz liste koja sadrži vremenske termine zauzetosti nekih sala (početak i kraj termina, broj sale) izbacuju oni koji se preklapaju; izbačene termine smestiti u novu listu;
4. određuje i izbacuje tačka najbliža sredini izlomljene linije predstavljene pomoću liste tačaka; pronađena tačka treba da deli liniju tako da je apsolutna vrednost razlike dužina linije pre i nakon te tačke minimalna;
5. izračunava površina konveksnog poligona (mnogougla) zadatog preko liste tačaka, a zatim se sa kraja liste izbacuje odgovarajući broj tačaka, tako da se novoformiranom poligonu površina smanji na 50-75% površine početnog ili tako da se poligon svede na trougao (izbacivanje prekinuti kad se ispuni bilo koji od ova dva uslova);

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 se kao argument dostavlja adresa pokazivača na početak liste i broj elemenata koji treba učitati, a preko povratne vrednosti vraća podatak o uspešnosti učitavanja. Potrebno je napisati funkcije kojima se vrši 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 i prekidanje programa. 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 se program testirati.** Odabrane test primere priložiti na listu papira pre odbrane. Kandidati koji ne 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 se pri pregledu programa lako može uočiti šta radi bilo koja programska celina. Takođe, poštovati pravila identacije određenih celina prilikom pisanja koda.

Važno: kada se reši osnovni zadatak, promeniti implementaciju ulančane liste tako da se koristi dvostruko ulančana lista. U skladu sa tim, potrebno je dodati odgovarajuće funkcije za ubacivanje i izbacivanje elemenata u listu, kao i sve ostale relevantne delove programskog koda. Takođe, potrebno je na početku i na kraju izvršavanja programa ispisati tačno vreme, a na kraju i vreme koje je utrošeno na izvršavanje programa. Opis funkcija za rad sa vremenom je u dodatku ovog dokumenta.

Napomene:

1. Odbrana četvrtog domaćeg zadatka je u RC ETF, u četvrtak, 7. juna, prema rasporedu koji se može videti na sistemu WebLab.
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 prototipe svih funkcija opisanih u postavci zadatka;
   - **dz4_unos.c**, **dz4_ispis.c**, **dz4_obrada.c**, koje sadrže izvorne tekstove potrebnih funkcija;
   - **dz4.dsp**, koja sadrži informacije o projektu koji sadrži sve potrebno za osnovni program;
   - **dz4_mod.c**, koja sadrži izvorni tekst modifikovanog programa na programskom jeziku C;
   - **dz4_ubl.dsp**, koja sadrži informacije o projektu koji se odnosi na dodatni program;
   - **dz4.dsw**, koja sadrži informacije o svim projektima relevatnim za ovaj zadatak.

30.05.2007. godine *sa predmeta*

Funkcije programskog jezika C za rad sa vremenom

Operativni sistem vodi evidenciju o iskorišćenom procesorskom vremenu za svaki od trenutno aktivnih procesa. U programskom jeziku C, ovo vreme se može dobiti pozivom funkcije **clock()** i izraženo je u "otkucajima" sistemskog časovnika (engl. *clock ticks*). Svaka implementacija jezika C podrazumeva sopstvenu vrednost simboličke konstante CLOKS_PER_SEC (u ranijim verzijama standarda CLK_TCK), koja predstavlja broj "otkucaja" sistemskog časovnika u toku jedne sekunde.

Sa druge strane, u svakom trenutku je moguće od operativnog sistema dobiti informaciju o vremenu u sistemskom časovniku, kako i o datumu i vremenskoj zoni gde se dati računar nalazi. Jednom kada se tačno vreme očita, ne postoji nikakva prepreka da se ono pretvori u odgovarajuću strukturu, odnosno prikaže na željeni način preko proizvoljnog znakovnog niza. Priloženi programski segment ilustruje prikaz tačnog vremena bez prikaza datuma:

```
time_t pIntTime = time(NULL);
struct tm* currentLocalTime = localtime(&pIntTime);
char* dateTimeString = calloc(100, sizeof(char));
/* if time setting and memory allocation was succesfull, create complete message */
if (currentLocalTime && dateTimeString)
  /* format the time as needed */
  strftime(dateTimeString, 99, "%H:%M:%S", currentLocalTime);
```

Dodatne informacije o ovde navedenim funkcijama (i svim ostalim koje se nalaze u zaglavlju **time.h**) se mogu naći u zbirci prof. Krausa, na **msdn.microsoft.com** i na stranici predmeta programiranje 2, koja sadrži dokument sa kompletnim ISO standardom za programski jezik C.