Primeri seminarskih zadataka iz ORS

Studenti formiraju zadatak po sopstvenom izboru, a u dogovoru sa asistentom. Priloženi primeri ilustruju složenost i obim zadataka za seminarski.

Zadatak SN1: Kalkulator

Napisati program *Kalkulator* koji izračunava računske izraze i grešku izračunavanja.

1. Napisati klasu *BrojG* koja predstavlja broj u dvostrukoj tačnosti (*double*) sa informacijom o tačnosti (tj. o grešci).
2. Za klasu *BrojG* napisati uobičajene matematičke operacije i funkcije (+, -, *, /, 1/x, sin, cos, tg, ctg, asin, acos, atg, actg, $\log_{10}$, ln, exp, x^y, sqr, sqrt, proizvoljan koren,...), tako da uz izračunavanje rezultata, na osnovu podataka o tačnosti argumenata izračunavaju i tačnost rezultata.
3. Napisati i metode za numeričko diferenciranje i integraciju proizvoljnih funkcija uz izračunavanje tačnosti.
4. Napisati program *Kalkulator* koji izračunava računske izraze pročitane sa standardnog ulaza. U izrazima se mogu pojavljivati konstante i uobičajene računske operacije i funkcije. Za konstante zapisane u uobičajenom obliku smatra se da je tačnost jednaka tačnosti podataka tipa *double*. Brojevi sa zadatom poznatom gornjom granicom apsolutne greške zapisuju se u obliku <*broj,greška*>. U izrazima se mogu pojaviti i konstante *e*, *pi* i *rez* (rezultat prethodnog izraza).

Literatura: predavanja i vežbe iz predmeta Uvod u numeričku matematiku.

Zadatak SA1: Merenje sjaja zvezda

Napisati program koji meri sjaj zvezda pomoću fotometrijskog snimka neba.

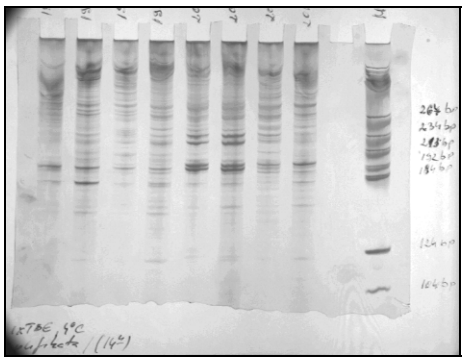
1. Napisati klasu *FSlika* koja predstavlja fotometrijsku sliku neba. Pretpostaviti da je dinamički raspon najviše 16 bita, tj. da je vrednost pojedinačnih piksela u rasponu 0 do 65535. Obezbetiti metode za čitanje slike iz datoteke i zapisivanje slike u datoteku u datom formatu (u prilogu).
2. Napisati metode koji obezbeđuju izračunavanje realnog dinamičkog raspona, tj. stvarnog opsega vrednosti piksela i otklanjanje šuma, tj. sjajnosti neba, i to: a) metod *minimax* koji pronalaza najmanju i najveću vrednost piksela, b) metod *histogram* koji deli taj raspon na 256 podoblasti i izračunava koliko se tačaka pojavljuje u kojoj od podoblasti, c) metod *fon* koji izračunava prosečnu fotometrijsku vrednost izmerenu kao fon neba.
3. Napisati klasu *SjajniObjekat* koja predstavlja apstrakciju sjajnih objekata na nebu. Svaki sjajni objekat ima svoj položaj (x i y koordinate slike), sjaj i relativnu širinu objekta. Aproksimirati sliku sjajnog objekta (nakon oduzimanja pozadinskog sjaja neba) normalnom raspodelom (u dve dimenzije). Relativna širina objekta predstavlja standardna devijacija te normalne raspodele. Položaj sjajnog objekta se izračunava kao srednja vrednost ove raspodele.
4. Napisati program koji pronalazi sjajne objekte na slici i meri njihov položaj, sjaj i relativnu širinu. Za sliku neba smatrati tačke sa relativnim intenzitetom do 8 % (omogućiti promenu ovog parametra). Opis svih pronađenih sjajnih objekata dopisati na kraj datoteke *sjajni.objekti.txt*. Program radi iz komandne linije i kao argument ima naziv (i putanju, ako je potrebno) datoteke sa slikom.

Literatura: bilo koji udžbenik verovatnoće i statistike za fakultet ili srednju školu; opis formata datoteke sa slikom, kao i primere slika neba tražiti od asistenta.

Zadatak SA2: Izdvajanje gelova

Skeniranjem gelova (ili fotografija gelova) dobija se slika poput priložene:

1. Napisati klasu *FSlika* koja predstavlja fotometrijsku sliku gela. Pretpostaviti da je dinamički raspon najviše 16 bita, tj. da je vrednost pojedinačnih piksela u rasponu 0 do 65535. Obezbetiti metode za čitanje slike iz datoteke i zapisivanje slike u datoteku u datom formatu (u prilogu).
2. Uz pretpostavku da je poznato da li se gelovi prostiru približno vertikalno i da na slici nema značajnih šumova (tj. da je na raspolaganju samo slika gela bez teksta, brojeva ili zapisanih oznaka) ustanoviti koliki je ugao odstupanja pravca prostiranja gelova od koordinatne ose.
3. Napisati metod koji za gel (pravougaonik čije stranice ne moraju biti paralelne koordinatnim osama) zadat koordinatama sredine početka i sredine kraja i širinom gela izračunava sumu intenziteta svetla po širini, tj. niz čiji elementi predstavljaju sume intenziteta po presecima gela širine jedne tačke. Dobijen niz predstavlja diskretnu funkciju koja predstavlja gustinu distribucije materijala u gelu.
4. Automatizovati čitav postupak pronalaženja zakrivljenosti, izdvajanja gelova, izračunavanja odgovarajućih diskretnih f.ja i snimiti dobijene podatke u potrebnom formatu.



Literatura: opis formata datoteke sa slikom i datoteke sa izveštajem, kao i primere slika gelova tražiti od asistenta.

Zadatak SG1: Geometrijske konstrukcije

Napisati klase koje omogućavaju izvođenje geometrijskih konstrukcija u ravni. Omogućiti da se u bilo kom trenutku neka od početnih pretpostavki izmeni, čime se menja kompletna konstrukcija.

1. Za svaki od osnovnih postupaka u konstrukciji formirati po klasu, i to najpre elementarne klase za objekte koji se mogu konstruisati bez osvrta na druge objekte (klase koje predstavljaju tačku, pravu, krug, poligon,...), a zatim i složenije klase za objekte koje se konstruišu uz upotrebu već konstruisanih objekata (prava kroz dve date tačke, krug kroz date tri tačke, tačka preseka pravih, krugova ili prave i kruga, prava kao tangenta na krug kroz tačku,...).
2. Obezbediti crtanje konstruisanih objekata. Omogućiti zumiranje, i posebno prikazivanje tako da na crtež staju sve tačke koje ne pripadaju pravim.
3. Omogućiti promenu položaja elementarno konstruisanih objekata, tako da se automatski ažuriraju svi ostali objekti.
4. Napisati program koji iz datoteke u datom formatu čita opis konstrukcije i crta je na ekranu. Omogućiti ažuriranje konstrukcije ako se izmeni sadržaj datoteke sa opisom konstrukcije.

Literatura: iz literature za predmet Osnovi geometrije izdvojiti osnovne postupke kojie se upotrebljavaju u konstrukcijama u ravni; od asistenta tražiti informacije o načinu crtanja u željenom režimu rada i o formatu za zapisivanje opisa konstrukcije.

Zadatak SF1: Redukcija λ-izraza

Napisati program koji izvodi redukciju λ-izraza:

1. Napisati klase koje omogućavaju predstavljanje λ-izraza u memoriji u vidu grafa. Neophodno je omogućiti pojavljivanje numeričkih i celobrojnih konstanti u izrazima, kao i uvođenje imenovanih izraza i određenog skupa predefinisanih imena.
2. Napisati metode za učitavanje λ-izraza sa standardnog ulaza i iz datoteke, kao i metode za ispisivanje λ-izraza na standardnom izlazu ili u datoteku.
3. Napisati metode za redukciju grafa λ-izraza.
4. Napisati program koji čita skup λ-izraza sa standardnog ulaza ili iz datoteke, vrši redukciju tih izraza i ispisuje rezultat redukcije na standardnom izlazu ili u datoteku.

Literatura: od asistenta i profesora tražiti dokumentaciju o λ-izrazima i redukciji grafa.

Zadatak SF2: Prevođenje λ-izraza u SK-kombinatore

Napisati program koji prevodi λ-izraze u SK kombinatore:

1. Napisati klase koje omogućavaju predstavljanje λ-izraza u memoriji.
2. Napisati klase koje omogućavaju predstavljanje SK kombinatora u memoriji.
3. Napisati program koji upotrebom osnovnih pravila za konverziju prevodi λ-izraze u SK kombinatore.

Literatura: od asistenta i profesora tražiti dokumentaciju o λ-izrazima, redukciji grafa i SK kombinatorima.

Zadatak SF3: Optimizacija SK-kombinatora

Napisati program koji vrši optimizaciu SK kombinatora:

1. Napisati klase koje omogućavaju predstavljanje SK kombinatora u memoriji.
2. Napisati program koji vrši optimizaciju SK kombinatora uvođenjem dodatnih složenijih kombinatora.

Literatura: od asistenta i profesora tražiti dokumentaciju o λ-izrazima, redukciji grafa i SK kombinatorima.

Zadatak V1: NAR

Napisati program *NAR* koji simulira NAstavni Racunar (NAR).

1. Napisati klase: *Memorija* koja predstavlja memoriju pomenutog fiktivnog racunara; *Procesor* koja predstavlja procesor tog racunara; koje predstavljaju instrukcije - MUA, AUM, SABF, ODUF, MNOF, DELF, PZAF, SAB, ODU, MNO, DEL, PZA, NES, NUS, BES, ZAR, POL, POD, KON, DIS, NEG, PIR.
2. Napisati metode za upis u memoriju i citanje iz nje; metode za upis i citanje akumulatora i indeks registara; metode za proveru korektnosti instrukcija kao i metode za izvrsavanje instrukcija.
3. Napisati program koji iz sekvencijalne datoteke ucitava program (omoguciti upotrebu svih vrsta adresiranja) a na standardnom izlazu prikazuje sadrzaj lokacija koje se navode iza kraja programa (u istoj datoteci u kojoj i program). Ime datoteke se moze zadati kao prvi aregument komandne linije ili se moze uneti po startovanju programa. U slucaju sintaksnih ili semantickih gresaka prikazati odgovarajucu poruku sa brojem reda u kojem se greska pojavila.

Literatura: Nedeljko Parezanovic - "Osnovi racunarskih sitema".

Zadatak VI1: Potapanje podmornica

Napisati program *Potapanje* koji omogucava odigravanje igre "potapanje podmornica".

1. Napisati klase: *Figura* koja predstavlja podmornicu i *Tabela* koja predstavlja tabelu u kojoj se odvija igra.
2. Napisati metode za rotaciju fugure, za postavljanje figure na zadatu poziciju, za otkrivanje polja, za utvrdjivanje da li je otkrivena cela figura, za utvrdjivanje da li su otkrivene sve figure, kao i ostale neophodne metode.
3. Napisati program koji ucitava dimenzije tabele, omogucava postavljanje pocetnog polozaja za oba igraca, a zatim obezbedjuje naizmenicno odigravanje poteza dva igraca i na kraju prikazuje pobednika.
4. Omogućiti snimanje trenutne pozicije i nastavljanje partije po učitavanju. Omogućiti i snimanje i reprodukovanje toka partije.

Zadatak VI2: Dame

Napisati program *Dame* koji omogucava odigravanje igre "dame".

1. Napisati klase: *Figura* koja predstavlja figuru i *Tabela* koja predstavlja tabelu u kojoj se odvija igra.
2. Napisati metod za proveru korektnosti poteza, za odigravanje poteza, za promociju figure u "damu", za proveru da li je igra zavrsena i ko je pobedio, kao i ostale neophodne metode.
3. Napisati program koji obezbedjuje naizmenicno odigravanje poteza dva igraca i prikazivanje pobednika na kraju igre.
4. Omogućiti snimanje trenutne pozicije i nastavljanje partije po učitavanju. Omogućiti i snimanje i reprodukovanje toka partije.

Zadatak VI3: Igra K Istih

Napisati program *IgraKIstih* koji omogucava odigravanje sledece igre: dva igraca upisuju svaki svoj znak u polaa kvadratne tabele (n x n). Na pocetku igre tabela je prazna. Igrac moze upisati svoj znak u prvo prazno polje (gledano odozdo) bilo koje kolone. Pobednik je onaj koji prvi upise k svojih znakova u susedna polja - bilo horizontalno, bilo vertikalno, bilo dijagonalno.

1. Napisati klase: *Figura* koja predstavlja figuru i *Tabela* koja predstavlja tabelu u kojoj se odvija igra.
2. Napisati metode za proveru korektonosti poteza, za odigravanje poteza, za proveru da li je igra zavrsena i ko je pobedio, kao i ostale neophodne metode.
3. Napisati program koji omogucuje zadavanje dimenzija tabele (n) i broja k, obezbedjuje naizmenicno odigravanje poteza dva igraca i prikazivanje pobednika na kraju igre.
4. Omogućiti snimanje trenutne pozicije i nastavljanje partije po učitavanju. Omogućiti i snimanje i reprodukovanje toka partije.

Zadatak V5: Recnik

Napisati program *Recnik* koji omogucava rad sa recnikom.

1. Napisati klase: *Pojam* koja predstavlja jednu rec u recniku (pojam) - obezbediti: zapis reci, njenu transkripciju, podatak o vrsti reci (imenica, glagol, pridev, …), znacenje reci, primer koriscenja, veza sa slicnim i suprotnim pojmovima i *Recnik* za rad sa pojmovima.
2. Za svaku reč potrebno je omogućiti evidentiranje njenih različitih formi (padeži, lica, vremena,...).
3. Napisati metode za dodavanje, promenu i brisanje pojmova recnika, metode za pretrazivanje recnika po pocetnom slovu, po slicnosti ili razlicitosti sa datom, po vrsti i znacenju (obezbediti znakove sa specijalnim

znacenjem: jedan (npr. *) koji zamenjuje bilo koji niz znakova i drugi (npr. ?) koji zamenjuje bilo koji (jedan) znak).

4. Napisati program koji omogucuje rad sa recnikom preko sistema menia. Recnik treba da bude napunjen sa bar 20 kompletno definisanih pojmova.

Zadatak V6: Karta

Napisati program *Karta* koji omogucava resavanje transportnih problema.

1. Napisati klase: *Mesto* koja predstavlja jedno mesto na karti (ime, broj stanovnika, povrsina), *Put* koja predstavlja jedan put (mesta koja spaja, duzina, cena transporta robe po jednom kilometru,) i *Karta* koja predstavlja sve puteve na karti.
2. Napisati metode za dodavanje, promenu i brisanje mesta na karti (nakon brisanja mesta azurirati duzine puteva), metod za proveru da li su dva mesta povezana (da li postoji nacin da se stigne iz jednog u drugo), metod za utvrdjivanje najkraceg rastojanja dva grada (uz prikaz mesta kroz koja treba proci), metod za utvrdjivanje najjeftinijeg transporta robe iz jednog mesta u drugo, metode za utvrdjivanje: mesta sa najvecim i najmanjim brojem stanovnika, najvecom i najmanjom povrsinom, najvecom i najmanjom gustinom naseljenosti, kao i mesta koja su najvece raskrsnice.
3. Napisati program koji omogucuje rad sa autokartom preko sistema menia..