Završni ispit 2013/14

Problemski dio

1. Razvijamo program za vektorsku grafiku koji podržava interaktivno uređivanje grafičkih objekata preko korisničkog sučelja. Pretpostavimo da program sadrži sljedeće komponente: model crteža (sadrži listu grafičkih objekata), pogled (odgovoran za iscrtavanje modela), te upravljač (odgovoran za prosljeđivanje korisničkih akcija primateljima). Korisničke akcije se ovisno o vrsti prosljeđuju ili grafičkim objektima (npr. rastezanje povlačenjem miša) ili modelu crteža (npr. postavljanje objekata u prednji plan). Prikažite organizaciju koja bi osigurala pravovremeno ažuriranje svih akcija na zaslonu računala.

2. Zadan je sljedeći izvorni kod u programskom jeziku C.

```
#include <stdio.h>
#include <stdlib.h>

struct Log;

typedef char const*(*PTRFUN)(

        struct Log*, char const*);

struct Log{

        PTRFUN* vtable;

};

struct LogFile{

        PTRFUN* vtable;
        FILE *pf;
};

//…

PTRFUN logFile[2] = {

         logFileAddEntry, logFileClose};
```

```
struct LogMail{

        PTRFUN* vtable;
        char const* email;
        char const* buffer;
};

//…

PTRFUN logMail[2] = {

         logMailAddEntry, logMailClose};

void test (struct Log *plog){

        plog -> vtable[0](plog, "proba 1");
        plog -> vtable[0](plog, "proba 2");
        plog -> vtable[1](plog, "kraj");
}


vodi main () {

        struct LogFile* p = logFileCreate (
                                "log.txt");
        test ((struct Log*) p);
}
```

Napišite definicije nedostajućih funkcija logFileAddEntry, logFileClose, logFileCreate. Skicirajte cjelokupnu implementaciju primjera u nekom objektno orijentiranom jeziku. O kojem bi se oblikovnom obrascu moglo raditi?

3. Dokument formata FRC pohranjuje slijed elemenata koji mogu biti ili odlomci teksta ili slike fraktala. Slike su predstavljene željenom veličinom te naputkom za generiranje kojeg može interpretirati komponenta FractGen za koju pretpostavljamo da je već razvijena. Problem je što generiranje slike može dugo trajati, a sam dokument može sadržavati više slika. Naš zadatak je predložiti organizaciju programa za pregledavanje takvih dokumenata koji bi osigurao
i) da se tekstovni dio dokumenta pokaže što je moguće prije
ii) da se za slike koje se nisu stigle generirati prikaže neka pretpostavljena grafika koja će automatski biti zamijenjena (i prikazana korisniku koji gleda taj dio dokumenta) po završetku generiranja slike.
U rješenju posebno obratite pažnju na strukturu razreda za pohranu informacija o dokumentu te na izvedbu „ubrzanog“ učitavanja dokumenta. Pretpostavi da je iscrtavanje potrebno implementirati u metodi OnPaint grafičke komponente MyCanvas u kojoj su već izvedene metode za prikazivanje odlomaka teksta i slika. Slika je modelirana razredom Image.

4. Razmatramo program za modeliranje 3D – tijela zadavanjem poligona koji omeđuju volumen tijela, pri čemu su (za sada) podržani samo trokut i pravokutnik. Poligoni se mogu po volji kombinirati: primjerice, proizvoljan peterokut se može prikazati s 3 trokuta. Jednom zadano tijelo korisnik mora moći jednostavno kopirati kako bismo pojednostavili zadaću generiranja scene koja sadrži više istih tijela.
Program treba podržavati i pomicanje tijela u prostoru. U razvijenom programu potrebno je podržati mnoštvo operacija nad tijelima scene: generiranje datoteke koja sadrži popis svih vrhova koji postoje u sceni, računanje ukupne površine oplošja svih tijela itd.

5. Oblikujte organizaciju programa koji omogućava pronalaženje optimuma funkcija definiranih nad podskupom R^n. Pretpostavi da se funkcije zadaju algebarski uz mogućnost korištenja podržanih pomoćnih funkcija npr. $f(x, y, z) = 3x^2 + 3\sin(y) + 2z^2$.
Program podržava uporabu različitih optimizacijskih algoritama (simulirano kaljenje, genetski algoritam, gradijentni spust). Nakon pokretanja optimizacije, program treba u grafičkoj komponenti prikazivati informaciju o trenutno pronađenom najboljem rješenju te mora podržati mogućnost zaustavljanja pretrage.

6. Oblikujte organizaciju aplikacijskog okvira GameAppFramework za izradu jednostavnih grafičkih igara. Pretpostavi da je za iscrtavanje u glavnom prozoru programa odgovorno „platno“ odnosno grafička komponenta razreda MyCanvas. Komponenta GameAppFramework izvedena je iz neke komponente koja predstavlja prozor (npr. JFrame). Prilikom pokretanja programa, korisniku se ispisuju podatci o igri te program čeka na pritisak tipke ENTER.
Nakon toga započinje konkretna igra koja se prikazuje za platnu i obrađuje korisničke akcije s tipkovnice i miša. Po završetku igre, program ispisuje poruku „Game Over“ te se na pritisak tipke ENTER vraća na početni ekran. Iz rješenja treba biti vidljiva organizacija aplikacijskog okvira te igre. Aplikacijski okvir ima metodu gameLoop koja određenom učestalošću poziva metodu render(MyCanvas c) odabrane igre. Uporabom opisanog aplikacijskog okvira korisnici bi morali moći jednostavno razvijati i igrati nove igre bez da vode računa o prethodno opisanim funkcionalnostima koje mora imati aplikacijski okvir.