Programmeringsprojekt

Brætspil

1 Introduktion

Formålet med dette projekt er at give deltagerne erfaring med basale GUI-mekanismer i Java og at lade dem praktisere deres generelle Java-kundskab.

Opgaven går ud på at implementere et brætspil, således at det kan spilles på en computer. Projektet udføres i grupper á 2-3 personer.

Der kan vælges mellem følgende spil:

1. Æselspil
2. Ataxx
3. Fanorona

Detaljerede beskrivelser af spillene kan findes via *projektsiden*:

`www.imm.dtu.dk/courses/02100/Project`

2 Problemformulering

Der skal udvikles et Java-program, der tillader det relevante antal spillere at spille spillet manuelt på en enkelt computer. Specifikt for 2-personers grupper skal programmet kunne:

- Vise spillets tilstand grafisk (bræt, brikker, tur, sidste træk osv.)
- Lade spillerne foretage deres træk på en bekvem måde
- Tjekke at spillets regler bliver overholdt
- Detektere spillets afslutning (og erklære en vinder)
- Gøre det muligt at gemme spillets aktuelle tilstand
- Indlæse og genoptage et gemt spil

For 3-personers grupper skal programmet yderligere gøre det muligt at:

- Gå frem og tilbage i det aktuelle spilforløb
- Gå tilbage til et tidligere trin og spille videre derfra

Programmet skal dokumenteres med en rapport som nærmere beskrevet i afsnit 6.

3 Programdesign

Programmet bør designes efter *Model-View-Control* princippet som vist i GUI-noten. Dette princip giver en god programopdeling, tillader systematisk test af indre dele og kan evt. benyttes som grundlag for arbejdsdeling.

Opgaven kan løses i en række trin:

1. Brætmodel. Fastlæggelse af datastrukturer til modellering af brættet og den aktuelle placering af brikker.
2. Træk. Definition af lovlige træk og deres ændring af modellens tilstand.
3. Grafik. Grafisk repræsentation af spiltilstanden.
4. Styring (eng. *control*). Fastlæggelse af, hvordan træk aktiveres/vælges og hvordan spilforløbet styres.
5. Ekstra funktionalitet (gemning/hentning osv.).

4 Afprøvning og demonstration

Det forventes ikke at der laves en intensiv test af alle programmets dele inden for kursets tidsramme, men som mininum kræves at I:

- Udvælger en central, ikke-triviel metode i modellen og foretager en grundig **strukturel test** af denne.
- Forbereder en **10-minutters programdemonstration** der dækker store dele af en funktionel test. Hertil skal der skrives en *demonstrationsplan*, der beskriver, hvordan programmet demonstreres. Planen skal specificere en række interaktionstrin, hvor der for hvert trin anføres:

  1. Formålet med trinnet, dvs., hvilken overordnet funktionalitet det skal vise (fx at det er muligt at lave et legalt træk).
  2. De handlinger der skal foretages (fx "klik på felt A3").
  3. De forventede reaktioner i GUI-en.

  Demonstrationsplanen skal inkluderes som et 1-2 siders bilag til rapporten og skal endvidere gives til hjælpelæreren **før** demonstrationen.
- Opsummerer de gennemførte tests af programmet og de konklusioner der kan drages heraf.

Nogle noter om GUI-test kan findes via projektsiden.

5 Nogle råd

- Det er **ikke** krævet at programmet skal kunne spille automatisk (som en eller flere af spillerrollerne). Hvis I kan finde en *simpel strategi* for automatisk spil, er det tilladt at tilføje dette som en ekstra facilitet udover det manuelle spil.
- I bør afholde jer fra at bruge Java-mekanismer, der ikke er gennemgået i kurset (fx træk-og-slip).
- Undlad at bruge tid og kræfter på at gøre grafikken "lækker", fx ved at bruge 3D grafik, animation eller lydeffekter. En sådan produktmodning kunne fx tænkes overladt til kvalificerede multimedie-designere.
- Brug ikke programudviklingsværktøjer til at bygge programmet med. Sådanne værktøjer vil ofte forsøge at skjule detaljerne omkring GUI-programmeringen, hvor formålet med dette projekt er netop at lære og forstå de basale GUI-mekanismer.

6 Rapportkrav

Programmet skal dokumenteres i en kort *rapport* med følgende indhold:

- **Forside.** Skal indholde samme information som for den første obligatoriske opgave.
- **Introduktion.** Kort, med angivelse af det valgte spil og formålet med programmet.
- **Problemanalyse.** Hvilke centrale begreber indeholder spillet? Er der nogen tvetydigheder eller uspecificerede tilfælde i spilbeskrivelsen? Hvordan kan problemet opdeles i underproblemer? Er der nogen specielt svære problemstillinger? Hvor kan forskellige løsningsprincipper tænkes anvendt, hvilke nogle er de, og hvordan har I valgt mellem dem?
- **Program design og implementering.** Hvad er den overordnede programstruktur? [Gerne illustreret.] Hvilke klasser er de vigtigste og hvad er deres ansvarsområder?

  Model-aspekter. Hvilke datastrukturer bruges til at repræsentere brættets tilstand? Har I overvejet alternativer? Hvilke algoritmer bliver brugt til at tjekke og udføre træk? Hvilket format bruges til at gemme og hente spil?

  Visnings-aspekter. Hvordan vises brættet og dets tilstand? [Skitser lay-out.] Hvordan opdateres grafikken? Hvordan laver man et træk via GUI-en?

  Styrings-aspekter. Hvordan styres spillets gang? Hvad sker der, når et træk er blevet aktiveret? Hvordan reagerer programmet på uventede hændelser? Hvordan hentes og gemmes spiltilstanden?

- **Afprøvning.** Hvordan er de forskellige dele af programmet blevet afprøvet? Hvilke typer test er der foretaget? Hvor godt virker programmet? Hvor grundigt er det blevet afprøvet?
- **Konklusion.** Hvad er blevet opnået (og hvad ikke)? Hvad har I lært om programmering af GUI-er? Hvordan kan programmet forbedres?
- **Bilag.** Se nedenfor.

Det anbefales at følge de stilistiske råd i noten *Writing Reports on Software*. Rapporten bør være på **højst 8 sider** (eksklusiv forside og bilag). Rapporten skrives på *dansk* eller evt. *engelsk*.

En *brugervejledning* (eng. *user's guide*) skal være inkluderet som bilag. Denne skal fortælle, hvordan programmet køres og hvordan træk og andre operationer udføres. Spillets regler kan antages kendt. Brugervejledningen forventes at fylde 1-2 sider.

Al *programkode* skal inkluderes som bilag på en sådan måde, at det er muligt at finde en bestemt klasse. Programteksten skal selvfølgelig være veldokumenteret som beskrevet i noten *Short Advice for Documenting Java Programs*.

Endvidere skal alle *tests og resultater* såvel som *demonstrationsplanen* optræde som bilag.

OBS: *Hver gruppe forventes at udarbejde program og dokumentation selvstændigt. Enhver brug af materiale fra andre kilder (andre grupper, www, ...) skal eksplicit angives i introduktionen med detaljer om samarbejde/oprindelse. I modsat fald vil sådan brug blive betragtet som forsøg på snyd og vil blive indberettet til studieadministrationen.*

7 Projektaflevering

Hver gruppe skal aflevere rapporten i én underskreven kopi senest **fredag d. 23. januar kl. 9.00** i postboksene i bygning 322's vestvendte trappeopgang. Samtidigt skal hver gruppe aflevere programmet elektronisk.

Samme dag, mellem kl. 9 og 12, skal hver gruppe demonstrere deres program over for en hjælpelærer.

Efter nærmere aftale vil det være muligt at aflevere og demonstrere torsdag d. 22. januar i stedet.

De praktiske detaljer omkring grupperegistrering, elektronisk aflevering samt demonstration vil komme til at fremgå via *projektsiden*. Ligeledes vil rettelser, uddybninger og FAQ til projektet kunne findes der. Væsentlige ændringer vil blive adviseret via CampusNet.

8 Evalueringskriterier

Opgaven vil især blive bedømt på basis af følgende kriterier:

- Analyse og beskrivelse af programdesignet.
- Teknisk kvalitet (kodning, kodedokumentation, afprøvning)
- Funktionalitet af brugergrænsefladen
- Rapportens stil og læselighed (formuleringer, stavning, grammatik, illustrationer)