Programmering og Problemløsning
Datalogisk Institut, Københavns Universitet
Arbejdsseddel 12 - individuel opgave

Jon Sporring og Christina Lioma

4. januar - 9. januar.
Afleveringsfrist: lørdag d. 9. januar kl. 22:00.

I denne uge fortsætter vi med at arbejde med objektorientet programmering, og vil fokusere på at opnå yderligere erfaring med at tænke objektorienteret.

Emnerne for denne arbejdsseddel er:

- Yderligere erfaring med klasser
- nedarvning
- og UML-diagrammer.

Opgaverne er opdelt i øve- og afleveringsopgaver. I denne periode skal I arbejde individuelt med jeres afleveringsopgaver. Regler for gruppe- og individuelle afleveringsopgaver er beskrevet i "'Noter, links, software m.m."'→"'Generel information om opgaver"'.

Øveopgaver (in English)

12ø0 Write a `Person` class with data properties for a person's name, address, and telephone number. Next, write a class named `Customer` that is a subclass of the `Person` class. The `Customer` class should have a data property for a unique customer number and a Boolean data property indicating whether the customer wishes to be on a mailing list. Write a small program, which makes an instance of the `Customer` class.

Afleveringsopgaver (in English)

Sporring, "Learning to program with F#", 2017, Chapter 21.4 describes a simplified version of Chess with only Kings and Rooks, and which we here will call Simplechess, and which is implemented in 3 files: `chess.fs`, `pieces.fs`, and `chessApp.fsx`. In this assignment you are to work with this implementation.

12i0 Extend the implementation with a class `Player` and a derived class `Human`. The intention is to prepare for a future derived class `Computer`, not to be implemented at the moment. The derived classes must have a method `nextMove`, which returns a legal movement as a codestring or the string "quit". A codestring is a string of the name of two squares separated by a space. E.g., if the white king is placed at a4, and a5 is an available move for the king, then a legal codestring for moving the king to a5 is "a4 a5". The codestring (for humans) is obtained by a text dialogue with the user.

12i1 Extend the implementation with a class `Game`, which includes a method `run`, and which allows two players to play a game. The class must be instantiated with two player objects either human or computer, and `run` must loop through each turn and ask each player object for their next move, until one of the players quits by typing "quit".

12i2 Make an extended UML diagram showing the final design including all the extending classes.

12i3 The implementation of `availableMoves` for the King is flawed, since the method will list a square as available, even though it can be hit by an opponents piece at next turn. Correct `availableMoves`, such that threatened squares no longer are part of the list of vacant squares.

Krav til afleveringen

Afleveringen skal bestå af:

- en zip-fil, der hedder `12i_<navn>.zip` (f.eks. `12i_jon.zip`)
- en pdf-fil , der hedder `12i_<(gruppe)navn>.pdf` (f.eks. `12i_jon.pdf`)

Zip-filen `12i_<(gruppe)navn>.zip` skal indeholde en og kun en mappe `12i_<(gruppe)navn>`. I den mappe skal der ligge en `src` mappe og filen `README.txt`.

I `src` skal der ligge følgende og kun følgende filer:

- chess.fs, pieces.fs, chessApp.fsx,

som beskrevet i opgaveteksten. Programmerne skal kunne oversættes med `fsharpc`, og de oversatte filer skal kunne køres med mono. Funktioner skal dokumenteres ifølge dokumentationsstandarden som minimum ved brug af `<summary>`, `<param>` og `<returns>` XML-tagsne. Filen `README.txt` skal ganske kort beskrive, hvordan koden oversættes og køres.

Pdf-filen skal indeholde jeres rapport ifølge:

- `Absalon->Files->noter->LaTeX->opgave.pdf`

guiden og oversat fra LaTeX. Husk at pdf-filen skal uploades ved siden af zip-filen på Absalon.

God fornøjelse.