Workshop 1 – Gruppe 17

*Første del:*

Emil Berglund

Abdisalam Elmi

John Olav Rabbevåg

*Andre del:*

Emil Berglund

Abdisalam Elmi

John Olav Rabbevåg

Ida Tollaksen

Max Oliver Solvang

**Første del:**

*Litt\_kode.cs*

**Hva gjør koden:**

*Public class file:*

Klassen reprensenterer en fil som består av flere deler (parts), hvor hver del er en streng. Man kan lage en tom instans, eller initialisere den med en liste av strenger. Klassens override returnerer hele listens innehold separert med komma mellom hver streng.

Konstruktørene: Parts (hoved), IEnumerable som parameter for å opprette liste.

Override: returnerer strenger med komma, imellom.

*Public class library:*

* Klassen lager et bibliotek med filer som legges i en liste.
* \_stuff lagrer File-objekter i en liste. Listen er privat.
* IReadOnlyList lines leser \_stuff og legger til i listen lines.
* IncludeFile(file F) legger til File-objekter i \_stuff.
* Metoden IncludeFile legger til en fil i biblioteket.

**Navngivning**:

Slik filnavnet er skrevet er feil. Ifølge retningslinjene skal klassenavn ha PascalCase. *Litt\_kode.cs* skal heller skrives som *LittKode.cs*.

*Public class file:*

Ifølge retningslinjer skal klassenavn ha PascalCase, og dermed være *File* fremfor *file*. Det samme gjelder *parts.*

*Public class library*

Klassens navn *library* skal ha stor forbokstav i følge retningslinjer. Feltet *stuff* er lite beskrivende, og bør i denne sammenhengen heller være \_*files.* Videre antyder *lines* at det er linjer som gås gjennom, når det egentlig er filer. *lines* burde derfor endres til *Files*. Metoden *IncludeFile* har riktig navngivning, men kan til fordel endres til *AddFile* for å bedre kommunisere hva den gjør.

*YEY!.cs*

**Hva gjør koden:**

Public class app: Klassen representerer en app for å kunne lese filer fra et bibliotek.

*Public library readstuff*

Koden tar i mot en filsti (place) som argument. Den sjekker deretter om filstien er tom eller null, og kaster passende exceptions deretter. Videre leser filen linje for linje med StreamReader, før den senere deler opp hver linje i mindre deler med *splitstuff*. Til slutt puttes de oppdelte linjene i en *file-*instans og legger dem til i en *library.* Returnerer et *library*-objekt som inneholder alle de oppdelte linjene fra filen.

*Private List<string> splitstuff*

*Splitstuff* har som oppgave å splitte strenger i mindre deler, med komma-separering.

Splitstuff(string chunk): Splitter strenger på komma i en liste.

En tom liste result for å lagre delene, looking som er en boolean variabel som bruker for å passe på splitting.

Bygger opp delene med StringBuilder i slice.

Item = “”, hvis looking er sann og neste felt er også tom, blir den lagt til tomrom og skipper indeksen.

Hvis looking er usann, (item = “,”) blir den lagt til i result, tømmer slice.

Siste slice blir lagt til i result.

**Navngivning:**

Filnavnet *YEY!.cs* bryter med retningslinjene på flere områder. Først og fremst skal filnavn ha PascalCase, og burde i såfall vært *Yey!.cs*, for det andre er YEY! Er lite beskrivende filnavn som ikke sier noe om hva filens innhold og formål er, for det tredje inneholder det spesialtegnet «!», som generelt burde unngås i filnavn.

*Public class app:*

Klassenavnet *app* er først og fremst i lower-case, noe det ikke skal være. *App* ville vært mer riktig her. Videre er *app* et lite beskrivende navn, som ofte betyr at klassen er en main-klasse som kjører et CLI e.l. Dette er ikke tilfelle her, og burde heller vært *CsvReader*, *FileProcessor,* eller noe i den duren.

Metoden *readstuff* er upresist og ikke i PascalCase. Navnet burde heller vært *ReadLibraryFromFile* for lettere forståelse og riktig navngiving.

Variabelnavnet *place* er upresist og vanskelig å forstå. Et bedre alternativ ville vært *filePath.*

Metoden *splitstuff* følger ikke retningslinjene for PascalCase, og burde heller vært *SplitStuff*. Videre er navnet upresist og gir lite innsikt i hva som splittes. Et bedre navn ville vært, *SplitLine.*

I metoden finner man også parameteret *chunk,* Dette navnet sier lite om hva objektet er. I stedet for *chunk*, burde det heller vært *line.*

Boolean verdien *looking* er også litt misvisende, og *insideQuotes* ville vært bedre her, for lesbarhet og forståelse.

*Prog.cs*

**Hva gjør koden:**

Klassen representerer et program for å lese data fra en fil (i dette tilfellet en .csv-fil).  
Path lagrer stien til filen med data.   
Det instaniseres et nytt app-objekt. Deretter brukes metoden readstuff fra app for å lese inn dataene. Hver linje leses inn, lagres som filobjekter og skrives ut i konsollen.

**Navngivning:**

Filnavnet *prog.cs* har flere feil som bryter med retningslinjene. Først og fremst er det lower-case. I tillegg er det forkortet. I dette tilfelle har defineringen av klassen det riktige navnet, mens filnavnet er forkortet og feil. Riktig navngivning er her, *Program.cs.*

Videre er det tidligere nevnte problemer som navngivning av *readstuff*. Det burde heller vært *CsvReaderProgram* eller liknende. I tillegg burde variabelnavnet *a* vært noe mer beskrivende.

**Andre del:**

Vi møtte gruppe 16.

Begge gruppene var ikke 100% ferdige da vi møttes 11:15, men vi fikk gått gjennom alt hvor vi gikk gjennom «prog.cs» sammen, da begge ikke rakk denne.

Totalt sett, var vi ganske samkjørte på det vi hadde funnet, og det vi mente programmene gjorde. Det samme gjaldt eventuelle forbedringer.

Totalt sett mente vi også at klassene til sammen fungerte som en CSV leser, og skriver, som i Main, skriver ut innholdet i det som er blitt skrevet til CSV filen.

Vi har dermed ikke noe særlig å melde her, annet at begge gruppene var enige om hva programmene gjorde.