**Bunden forudsætning**

Adventure del I - refactored

I har fået koden til Adventure del I til at virke. Men før I kan begynde at tænke på del II, skal der ryddes op.

Koden skal refaktoreres – altså funktionalitet skal flyttes, uden ellers at ændres – og der skal benyttes nogle gode design-principper, så det bliver nemmere at arbejde med koden fremover.

Det betyder at der skal laves nogle flere klasser – en der tager rollen som brugerflade, en der påtager sig at være controller, en der er creator, og eventuelt endnu flere for at sikre Single Responsibility Principle.

Krav

Controller (Adventure)

Adventure klassen skal være “controller”, altså den klasse der styrer hele programmets flow, og som koordinerer samarbejdet mellem objekterne i programmet.

Single Responsibility Principle (UserInterface og Adventure)

Der skal være én klasse til at håndtere al user-interface – det er den eneste klasse der må have metoder til udskrift og indlæsning. Alle andre klassers metoder returnerer data eller modtager parametre.

Adventure klassen kører selve spillet – den håndterer det der rent faktisk sker i spillet. Når brugerfladen har fortolket hvad brugeren har indtastet, så er det Adventure, der sørger for at sende det videre.

Creator (Map)

Der skal være én klasse (Map) der opretter rummene, og kæder dem sammen – altså bygger hele spillets map. Den klasse kaldes af controlleren, inden spillet går i gang.

Adventure vs Player

Adventure-klassen styrer selve spillet, men det bør ikke være dens opgave at holde styr på hvor “spilleren” er, det må “spilleren” selv gøre. Så I skal have en spiller klasse (Player) og hermed et spiller objekt , der kender spillerens position på spillepladen (hvilket rum spilleren er i). Det er også det objekt der bør håndtere at flytte “sig selv” rundt, og tjekke om en ønsket retning overhovedet er mulig.

Navngivning

Brug gode sigende navne til klasserne – ord som controller og creator er alt for generiske. Brug nogle navne der relaterer til jeres Adventure-spil, og mere fortæller, den der læser koden, hvad den pågældende klasse gør, end hvilken software designmæssig rolle den påtager sig.

Klassediagram

For at få overblik over jeres nye programdesign, skal I lave et komplet klassediagram. Det skal tegnes på computer – tegnes, IKKE autogenereret fra IntelliJ.

Repository

Fortsæt med at kode i det samme repository. Sid sammen og lav ændringerne – det er altid en dårlig ide at dele den slags oprydning op mellem sig!

Aflevering

Hvordan

Gen-aflever blot et link til repositoriet, når I er færdige.

Upload også en pdf med klassediagrammet.

Hvornår

Helst i dag, men bare inden I begynder at arbejde på Adventure del II.