Adventure del 5  
  
I skal arbejde på afslutningen af adventure-spillet. Spilleren kan bevæge sig rundt, samle ting op og efterlade dem igen, nogle ting er mad der kan spises, og andre er våben der (indtil videre) kan bruges til at angribe den tomme luft.

Nu skal I udvide spillet så der også er fjender som man kan angribe med våbnene! Og som kan angribe tilbage, så man mister health, og må spise noget.

Krav 

Dokumentation   
  
Som noget nyt (i forhold til de tidligere dele) skal der også afleveres dokumentation til denne afsluttende version. Dokumentationen skal indeholde:

* Et klassediagram over alle klasser i spillet, deres arveforhold og associationer (inkl. multiplicitet).
* Et aktivitetsdiagram over attack-sekvensen .

Se mere i afsnittet om “Anbefalet procedure” for hvordan denne dokumentation kan skabes. 

Spillet 

Der skal også være fjender i rummene. Fjenderne er ikke ting der kan samles op og bæres med, men de er heller ikke så levende at de kan flytte sig fra rum til rum – de kan blot angribe eller angribes.

Spilleren skal have en form for health, som kan øges ved at spise (sund) mad, og mindskes ved at blive angrebet af fjender.   
  
Brugerfladen

Spillet skal ikke udvides med flere kommandoer, men attack skal ændres:

attack

eventuelt efterfulgt af navnet på en fjende i rummet. Det aktuelle våben bliver så brugt til et angreb på den nævnte fjende, alternativt den fjende der er nærmest.

Attack er nu mere kompliceret end nogen anden kommando – hvis der ikke angives et navn, angribes den nærmeste fjende i rummet; hvis der ikke er nogle fjender i rummet, angribes den tomme luft. Men for eksempel skydevåben har et begrænset antal skud i sig, og prøver man at angribe med et tømt våben, skal man have at vide at det mislykkes. Hvis man ikke har et våben “equipped” skal man også få at vide at det mislykkes.

Attack af fjender er endnu mere kompliceret – så her følger en detaljeret gennemgang af hvad der skal foregå: 

Først angribes fjenden med det våben som spilleren har equippet. Fjenden mister health svarende til den damage våbenet giver.

Derefter angriber fjenden spilleren – det sker med det samme, og spilleren kan ikke nå at flygte ud af rummet, selv ikke hvis der er angrebet med et langdistance våben. Fjenden er også udstyret med et våben, og spilleren mister health svarende til den damage dét våben giver.  
  
Forudsat at begge parter stadig er i live, er attack-sekvensen sådan set ovre – og spilleren kan vælge at gå ud af rummet eller skifte våben, eller attack’e igen. Fjender angriber ikke uprovokeret (i hvert fald ikke i grundversionen).

Hvis fjenden mister al sin health, dør vedkommende, og drop’er sit våben (som spilleren efterfølgende kan samle op), og forsvinder selv fra rummet – måske efterlader den et lig i form af et item, som spilleren også kan samle op.   
  
Dette er den grundlæggende attack-sekvens – I er velkomne til at gøre den mere avanceret :)

Derudover skal brugerfladen udvides, så man sammen med beskrivelsen af et rum, får en liste over eventuelle fjender, og når man træder ind i et nyt rum, skal man som minimum have at vide om der er fjender i rummet.

Koden

Enemy skal være en klasse helt for sig selv, og altså ikke arve fra Item – Room skal have en særskilt liste til enemies, med sine egne add og get metoder.

*Enemy*

Enemy objekter er ikke Items, men deres helt egne!

En enemy skal have et navn, en beskrivelse, et health-niveau og et enkelt weapon. Til forskel fra player, har en enemy altid sit eneste våben equipped, og kan ikke skifte mellem våben.

En enemys våben er et almindeligt Weapon objekt, som kan overtages af playeren når enemyen er død, så det skal kunne droppes i rummet.   
  
Enemy skal, ligesom Player, have attack og hit metoder, for henholdsvis at angribe (attack) player med sit våben, og blive angrebet (hit) af players våben.

Enemy skal selv opdage om den er død, droppe sit weapon, og forsvinde fra rummets liste over enemies, samt eventuelt efterlade et item (sit lig).  
  
   
Anbefalet procedure

1. Lav **attack** kommando’en uden at tage en fjende som parameter, men så spilleren blot attacker det tomme rum, og for eksempel affyrer et skud (hvis det er et RangedWeapon)  – den del har du nok allerede lavet i forbindelse med Weapon
2. **Tegn et aktivitetsdiagram** for hvad der skal ske i **attack-sekvensen**: Tænk på hvordan og hvornår der skal tjekkes om player overhovedet har et våben equipped, om våbenet kan bruges, altså har skud tilbage (husk, ingen test på om det er et RangedWeapon, Weapon objektet skal blot selv fortælle om det stadig kan bruges). Tænk også på hvilke beskeder der skal gives tilbage til spilleren, for eksempel om der overhovedet er et våben, om våbenet stadig er brugbart, om det var det sidste skud der blev brugt, om man ramte en fjende eller det tomme rum, om hvordan fjenden reagerer, om hvad der sker hvis fjenden dør, hvad der sker hvis man selv bliver ramt, etc. etc.   
   Der er meget at tage hensyn til – så tegn og diskutér **før** I koder!   
   Aktivitetsdiagrammet må gerne tegnes på papir.
3. **Kod attack-sekvensen**

Frivillige forbedringer og udvidelser

Det er svært at tro at der bliver tid til udvidelser, men skulle I få attack-sekvensen til at fungere, og sidde med god tid til overs, så har vi en masse forslag og ideer til udvidelser. De er beskrevet selvstændigt i  [Adventure del 5 - frivillige udvidelser](https://kea-fronter.itslearning.com/ContentArea/ContentArea.aspx?LocationID=6767&LocationType=1). I er selvfølgelig også velkomne til at implementere nogle af udvidelserne fra de tidligere versioner, eller finde på jeres helt egne. Husk blot at committe den fungerende udgave først, så I altid kan rulle tilbage, hvis ideerne bliver for vilde.

Aflevering

I må stadig gerne arbejde i jeres studiegrupper, men husk at I (som tidligere) afleverer individuelt. 

Hvad

Der skal afleveres en pdf med følgende:

* Forside der indeholder:
* Navnet på spillet
* Et fængende cover-billede til spillets æske
* Link til GitHub repository til koden – sørg for at linket både er klikbart, og udskrevet så det kan printes ud og tastes ind. (Det GitHub repository må gerne være den som I har arbejdet løbende med.)
* Navne og github-brugernavn
* Klassediagram over alle klasser, associationer og arveforhold
* Aktivitetsdiagram over attack-sekvensen

Klassediagrammet **skal** være det gældende klassediagram for det færdige produkt – aktivitetsdiagrammet er et designdokument, og behøver ikke nødvendigvis afspejle eventuelle ændringer i koden. UML diagrammerne kan ligge i GitHub repo (i en docs folder).

Hvordan

Upload pdf’en som besvarelse på denne opgave.

Hvornår

Senest d. 10/10 efter undervisningen.

Feedback

Studiegruppepræsentationer af udvalgt kode og diagrammer. Nærmere information følger!