**Vad har jag lärt mig under utvecklandet av Snake?**

Det första jag konstaterar är att det enklaste sättet att bygga Snake på är att använda sig av Arrays för att flytta delar av ormens bakre del och sätta den längst fram.

Exempel. Om detta är ormen \*\*\*\* så flyttar du hela tiden den bakre stjärnan och lägger den längst fram och

På så sätt flyttar ormen.

Du orsakar inga problem med minnet och programmet kraschar inte som innan när jag använde mig av en Tick-metod() att hela tiden använda mig av en If-sats och flytta fram ormen.

Efter ett OÄNDLIGT antal tutorials och tester av olika varianter, få Java.swing och java.awt och funka som jag vill så har jag hittat någon typ av lösning.

Efter många timmar på Google så hittade jag ett simpelt program för snake med ganska lite kod och endast 2 klasser. Det intressanta med den var att han hade byggt upp ormen på ett sätt som jag själv att tänkt mig och likt den koden Joakim visade mig i sitt Snake som var gjort i Python.

Jag tog inspiration därifrån och försökte göra mitt eget program med 5 klasser och byggde det på ett liknande sätt. Personen var en tysk så fick försöka förstå hur han hade tänkt i sin kod eftersom jag inte förstod hans text.

***FAS 1***

* *Planering och research om Javas egna GUI JFrame, JPanel. Eftersom jag gjort ett simplare PingPong-spel tidigare så var det smart och använda ett liknande koncept.*
* *Strukturera upp klasserna. Screen – Snake – Game – Grid – Score. Jag struntar nu i att göra en egen klass för Apple då jag bakar ihop den i en Enum inner class som jag döper till Type som skall innehålla APPLE, WALL, SPACE. SPACE blir planens yta.*
* *Sätta storlek och få upp skärm*
* *Måla upp Grid medhjälp av en tvådimensionell Array (x och y).*
* *Börjar göra en UML karta för träningens skull och en liten överblick på hur jag vill att det skall se ut. Även om den kanske inte är helt korrekt gjord.*

***FAS 2***

* *Implementerar Interface KeyListener för att kunna ställa in styrning med tangenter.*
* *Implementerar Interface Runnable för metoden Run.*
* *Ställer in KeyEvent och kodar upp en switch-sats då den är smidigare i en flervalssituation.*
* *Använder mig av en LinkedList istället för en ArrayList då många andra verkar föredra det och den blir lättare och koda för min del.*
* *Efter att vi går igenom Highscore list på lektion så använder jag den koden som ett startskott till mitt spel. För att få det grafiskt snyggare så använder jag mig av JOptionPane från min gameOver metod.*

***FAS 3***

* *Klassen Game blir där allt får utlopp och där allt knyts samman. Använder mig av Java Timer och Timertask som thread vilket jag får och funka bättre än när jag använde mig av klassen Thread i tidigare försök.*

***SPELET KLART:***

* *Starta spel och ormen rör sig när du tryckt i första tangenten med vilken riktning du skall börja.*
* *Snake förflyttar sig*
* *Äpple slumpas ut på spelplanen*
* *Ormen växer varje gång han äter äpple*
* *Poäng räknas från 0 och ökar med 1 vid varje äpple du plockar*
* *Spelet avbryts om du krockar med dig själv, med väggen eller trycker tangenten som är i motsatt riktning från dit du är på väg.*
* *Spelet visar att det är över då det kommer upp en ”Game Over” prompt vid död.*
* *Spelet får upp ännu en prompt där du skriver in ditt Score.*

***SPELETS BRISTER:***

* *Highscore-listan byggs bara på och puttar inte ut det sämre resultaten. Så varje spelare får sätta sitt resultat i textdokumentet och så får man radera det manuellt.*
* *Spelet uppdaterar sämre ju större ormen blir.*
* *Försökte på till en simpel meny med ”Play”, ”Quit” i spelet men det har jag fått ihop ännu.*

***SAMMANFATTNING***

*Ett mycket svårare projekt än jag hade först trott då det verkligen gäller att tänka till om vilket riktning du skall ta. Det finns så många olika sätt att göra spelet mina tre första planer gick inte som jag ville så fick jag göra ännu mer research framförallt mixtra med olika if-satser och få till mina arrays [][] för det är där den stora utmaningen i att göra Snake ligger. Samtidigt skall man använda många – för mig - helt nya delar så som Linkedlist, (till viss del Enums), Timer thread, och även GUIn.*

*Jag har försökt ta med mig saker från tidigare föreläsningar eftersom det är det vi skall kunna visa att vi förstår, samtidigt som jag experimenterat lite själv.*

*Tvådimensionella arrays, konstruktorer, metoder, super- och subklasser i arv har jag fått med och utvecklat ett spelbart spel som är lite bättre än orginalet.*