OO-modeller

Joachim von Hacht

OO-model for a Harbour

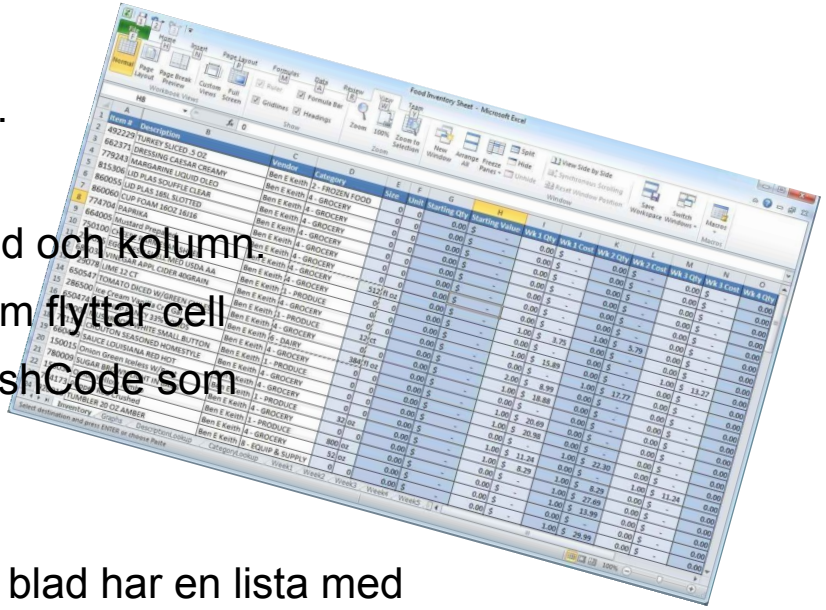
1. Create class for at ship. Ship has id, length and depth.
   We assume any setter/getter/equals or hasCode methods are there, don't need to implement). All data set when objects created.

2. Create class for a QuayPlace A "place" where a ship can berth (lägga till). QuayPlace has length, depth, possible a ship and an id. All data except ship set when QuayPlace created. Add a method that let a ship berth at the quay place, and a method isOccupied() returning true if there is a ship at the quay.

3. Create class Quay. Quay has many QuayPlaces. Add method to tie a ship to the some QuayPlace. If success return true else false.

4. Create class Harbour. Has many Quays.

OO-model for an Excel sheet



Skapa en OO-modell för ett kalkylprogram typ Excel.

1. Skapa klass för en cell i ett kalkylblad. Cell har rad och kolumn. Båda sätts då cellen skapas. Lägg till en metod som flyttar cell ett steg till höger. Vi antar att get/setters/equals/hashCode som kan behövas finns (gäller även nedan).

2. Skapa en klass för ett kalkylblad (med celler). Ett blad har en lista med celler samt max antal rader och kolumner. Rader och kolumner sätts då bladet skapas. Då bladet skapas skapas även alla celler i bladet.

3. Lägg till en metod i bladet, som givet ett kolumnnummer, lägger till en ny kolumn (med tomma celler) till vänster om kolumnnumret (insert new column left).

OO-model för “That’s Life” (ett spel)

Se https://boardgamegeek.com/boardgame/17240/s-life

Klasserna skall vara så icke-muterbara som möjligt och dölja så mycket som möjligt av sin data (information hiding).

1. Skriv en klass Tile för brickor. En bricka har ett visst antal poäng. Poäng anges då man skapar brickan. Vi antar att get/setters/equals/hashCode som kan behövas finns (gäller även nedan).

2. Skriv en klass Player för en spelare. En spelare har ett namn och en lista med brickor (spelaren kan samla på brickor). Det skall finnas en metod som lägger till en bricka i spelarens lista. Namn anges då man skapar en spelare.

forst. nästa slide

forts.“That's Life”

3. Skriv en klass Piece som skall representera en spelpjäs. En Piece har en färg, en spelare som äger den och en position, d.v.s. den Tile den står på. Alla data skall anges då pjäsen skapas.

4. Skriv en klass Game för hela spelet. Game har en spelplan i form av en lista med Tiles, en lista med Piece för pjäserna samt en aktuell pjäs (den som skall flyttas). All data sätts då ett Game-objekt skapas.
   Lägg till en metod move(int diceResult) i Game. Metoden flyttar aktuell pjäs enligt parametern diceResult. Om det efter förflyttningen är tomt på brickan där pjäsen stod skall metoden ta bort brickan från spelplanen och tilldela denna till spelaren som äger pjäsen (spelplanen krymper)