Textuppgift 1

Motivering av valet av Interface eller generalisering för att hantera olika typer av skepp  
  
Vi valde att använda oss av ett interface istället för generalisering. Detta valet gjorde vi efter som att samtliga skepp delar i princip samtliga metoder som, skall implementeras. Ett interface gjorde detta enklare då metoderna kan vara fullständigt färdigställda i kontrast till en abstrakt klass där metoderna implementeras i respektive subklass.

Hantering av två spelare

Ett sätt att utöka programmet för at hantera flera spelare hade kunnat vara att rita ut två spelplaner samtidigt genom att skapa en till MainFrame. Därefter hade respektive spelare fått välja vilket upplägg de vill ha på sin spelplan. Vilken spelare som börjar hade sedan slumpats fram. Därefter frågar spelet spelare 1 att välja en ruta på spelare 2s spelplan, under tiden som detta sker är hela MainFramen för spelare 1s spelplan ”Disablead” och spelaren kan därför inte klicka på fel spelplan. När en spelare har vunnit meddelar spelplanen där en av spelarna har genomfört det slutgiltiga draget Controller-Klassen att den har vunnit och därefter kan Controller-klassen meddela MainFrame för den andra spelplanen att den har förlorat och skriva ut ett meddelande.

Frågor för andra gruppen

Vi hade ett problem med flödet i programmet där vi skrev ut en träff på den vinnande rutan efter att spelaren vunnit och startat en ny runda. Hade ni detta problem och hur löste ni det.

Vi hade svårt att välja hur vi skulle spara namnen för de olika skeppen och landade slutgiltigen i att använda en Enum för detta. Hur valde ni att spara namnen för de olika skeppen, vilka för och nackdelar finns det med metoden som ni valt?