Dokumentation - Brädspelet
Grupp 2 - Swedish Test Mafia
Programmering för testare HT24, Labb 2

Hur man spelar spelet

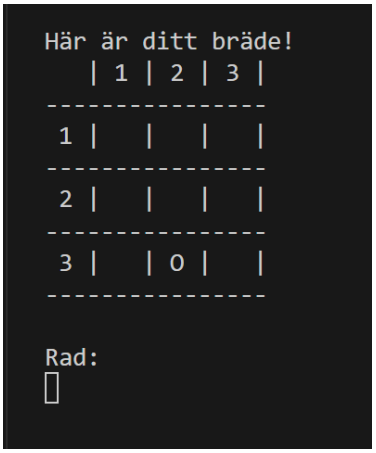
När programmet startas möts spelaren av en meny med fyra val där de kan välja att spela följande spel:

1. Spela tre i rad mot datorn
2. Spela tre i rad mot en motståndare
3. Spela fyra i rad mot en motståndare
4. Spela fem i rad mot en motståndare

Därefter får spelaren välja om den vill skriva in ett eget namn eller behålla ett default-namn, *Spelare 1 / Spelare 2*.

Spelaren får sedan välja mellan symbol ‘X’ eller ‘O’, som de vill spela med. Därefter slumpas det fram vem som får inleda omgången, oavsett om spelaren spelar mot datorn eller mot en annan motståndare.

Brädet presenteras sedan i terminalen där rader och kolumner är presenterade med siffror. Spelaren får sedan först skriva in vilken rad och sedan vilken kolumn som de vill placera sin symbol. (Se bild nedan.)



Därefter får spelarna lägga en symbol varannan gång och målet med spelet är att placera sina symboler i tre/fyra/fem i rad (beroende på vilket spel som valts). Om spelaren försöker placera på en upptagen ruta eller på en ruta utanför spelplanen, presenteras ett felmeddelande. Felmeddelandet informerar spelaren vad som blev fel, och erbjuder ett nytt försök.

Vinstvillkor

Om någon av spelarna har fått tre/fyra/fem i rad (lodrätt, vågrätt eller diagonalt) avslutas omgången och spelet presenterar vinnaren. Om brädet blir fullt avslutas omgången och spelet presenterar att omgången är oavgjord.

Efter avslutad omgång presenteras spelaren med tre val:

1. Börja ny omgång
2. Tillbaka till menyn
3. Avsluta spel

Poängöversikt

Vid avslutad omgång presenteras spelarnas vinster för aktuellt spel, samt spelarnas vinster totalt sedan spelet startade.

Uppbyggnad och val av kodstruktur

Vi har valt att utveckla fyra spelvarianter som har nästan identiska spelflöden. Det som skiljer dem åt är storlek på bräde, antal symboler i rad för vinst, samt om ‘spelare 2’ är en dator eller inte. Vi har variabel *gameID* i klassen *Main* som vi använder för information om detta. Spelflödet i klassen *Game* är därmed återanvändbart för samtliga spel.

Spelbrädesstatus (vinst/oavgjort, skapande av bräde, printfunktion m.m.) hanteras i klassen *Board* och är oberoende av spelbrädets storlek.

För att hantera spelbeteende om spelaren är en dator, har vi lagt till instansvariabeln *isHuman* i klassen *Player*.

För kontroll av integer-input har vi en klass *GlobalTools* med ett inputfilter som endast släpper igenom giltiga värden. Val av spelarnamn och felhantering därav hanteras i klassen *Player*.

För gruppering av aktiva spelare använder vi oss av en *ArrayList*, för att kunna lägga till och ta bort spelare, samt sätta spelare som “vilande” om spelaren växlar spelläge vs dator eller vs motspelare. Och en Queue där vi lägger en slumpmässig turordning.

Vi har lagt *Player* som en separat klass för att göra instanser och hantera respektive spelares namn, symbol och statistik.

Brädet kan göra i olika storlekar, men vi har valt att förbestämma brädstorlek även för spelen fyra i rad och fem i rad.

Klasser

Main

Klassen *Main* har variablerna *gameScanner* och spellistan *players* som används globalt i hela spelet. Klassen har även spellistan *restingPlayers* där spelare som inte spelar just nu "vilar" samt variablerna *activeGameID* och *nameForPlayer2IsSet* som används för att kolla om vi behöver skapa en ny spelare när 'spelare 1' byter spelläge från "vs dator" till "vs motspelare".

Här startas spelet och använder klassen *Menu* för att visa en meny med de olika spelvalen. Spelaren gör sitt val och en spelarlista skapas. En instans av klassen *Game* skapas.
Har metoder för att skapa en spelarlista, skapa och hantera en spelarlista för "vilande spelare" (om spelare 1 vill spela mot datorn i stället för motståndare och vice versa), samt skickar oss vidare till själva spelomgången.

Menu

I klassen *Menu* finns meny för val av fyra olika spellägen, *Tre i rad mot dator, Tre i rad mot motståndare, Fyra i rad mot motståndare* och *Fem i rad mot motståndare*. Menyn är uppbyggd av en switch-sats med felhantering om spelaren matar in ogiltiga symboler/siffror. I menyn är olika spelen kopplade till olika ID:n som är unika för varje spelläge.

Player

Klassen *Player* har instansvariablerna *name, symbol, isHuman* och *stats*. Här finns metoder för att välja namn på spelaren/spelarna, felhantering vid inmatning av namn, och hantering av spelarens vinststatistik. Det finns även en metod som behandlar att spelaren ska kunna välja vilken symbol de vill spela med.

Game

Klassen *Game* har instansvariablerna *gameID, gameOn, playOrder, curentPlayer, gameBoard* och slumpgenerator *rand*. Här hanteras spelflödet för de olika spelen. Denna klassen använder metoder från *Player*-klassen för att hämta information om aktuell spelare, och från *Board*-klassen för att skapa ett spelbräde, kontrollera brädets status, placera symbol och skriva ut aktuellt bräde i terminalen. Det finns metoder som slumpar ordningen på spelarna, byter spelartur, hanterar vinstkriterier, hanterar dators beteende om motspelare skulle vara en dator, samt vad som händer vid avslutad omgång (en meny med val att spela igen, avsluta spel eller gå tillbaka till startmeny). Ett välkomstmeddelande och med instruktioner skrivs ut vid start av varje spel och ett avslutningsmeddelande vid spelavslut.

Board

Klassen *Board* har instansvariablerna *name, isFull, table, spacesTaken* och *spacesTotal*. Här hanteras spelbrädet, med metoder för att skapa ett bräde, skriva ut brädet i terminalen, kontrollera om en plats är giltig (finns på brädet) samt om en plats är ledig. Även metoder för

att placera en symbol samt kontrollera om vinstvillkor uppfyllts (om brädet har tre/fyra/fem i rad).

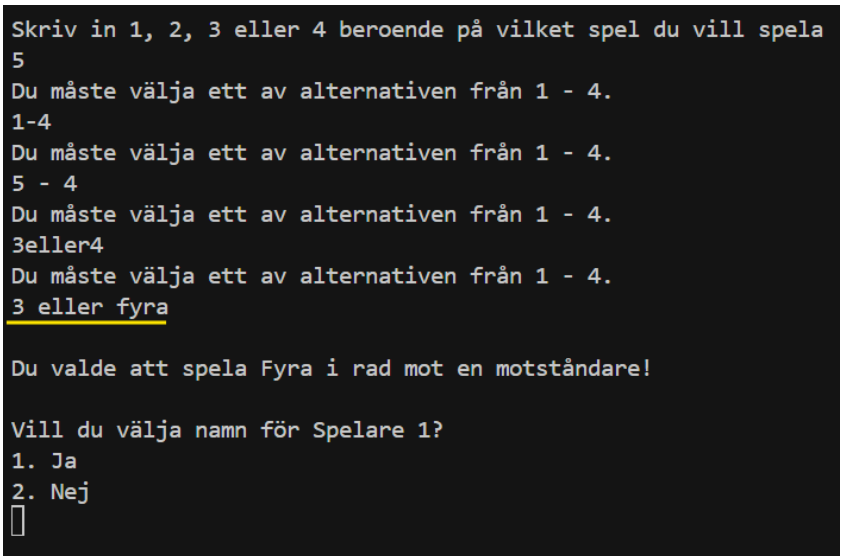
GlobalTools

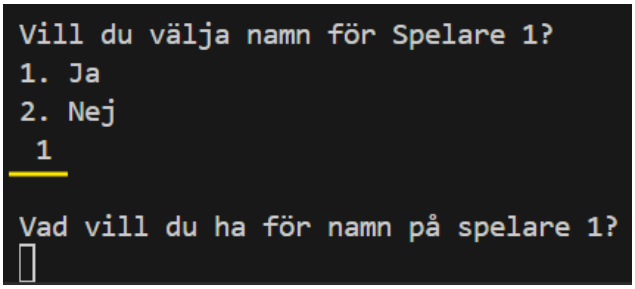
GlobalTools innehåller ett inputfilter som endast släpper igenom giltiga värden vid användarinput. Används i klasserna *Main, Menu, Game* och *Player.*

Kända fel/buggar

Vi upptäckte en bugg som berör inputet av valalternativ, till exempel "*Du måste välja ett av alternativen från 1-2*", och "*Välj ett spel mellan 1-4*". När spelaren matar in sitt numeriska val exempelvis "*1 - 2*", "*3 eller 4*", eller matar in en siffra som uppfyller villkoren med följt av mellanslag och andra tecken, läser spelet första integern som skrivs (se bifogad bild nedan). Förväntat resultat vore att felmeddelandet skulle promtas igen för spelaren, istället för att acceptera och tillåta spelaren gå vidare. Om inmatning inleds av ett mellantecken först innan spelaren skriver in ett svar som uppfyller villkoren, läses också in, vilket inte bör göras.

Med anledning av tidsbrist valde vi att inte åtgärda buggen då vi bedömde den som icke kritisk. Vi tar dock med oss det som lärdom inför framtida uppdrag.





Tillståndsdiagram