# INDU: Tre i rad

8 december 2021

*Detta projekt är utvecklat av Kristoffer Sahlin med små modifieringar av Anders Mörtberg.*

## Innehåll



## Introduktion

I det här projektet ska du skriva ett Pythonprogram för att låta datorn spela tre i rad genom simuleringar. Projektet består av två uppgifter. För att kunna få betyg E-C räcker det att göra uppgift 1. För att kunna få högre betyg än C måste man ha gjort båda uppgifterna. För generella krav på projektet samt betygskriterier se de generella instruktionerna i *INDU—Individuell Uppgift i Programmering*.

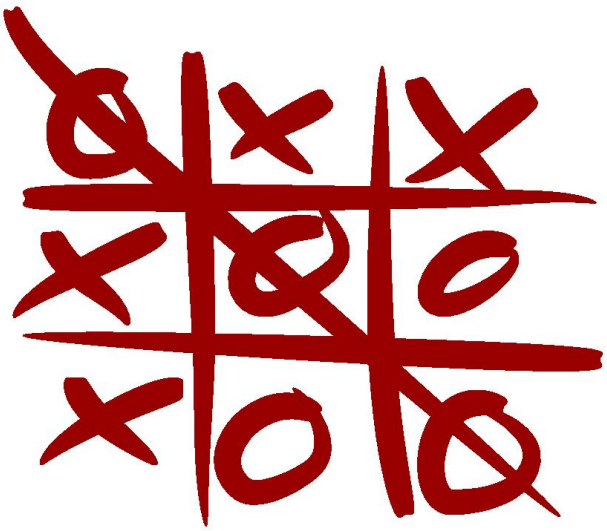
## Regler för tre i rad

Det finns olika varianter på tre i rad. Vi ska spela en version där man har en fördefinierad storlek på brädet $k \times k$ och lägger till markörer på brädet tills det är fullt (exempel i fig. 1). Till skillnad från det traditionella tre i rad med två spelare ska vi anta att **tre spelare** spelar samtidigt. Spelarna turas om att sätta sin markör på brädet i ordningen spelare1, spelare2, spelare3, spelare1, och så vidare. Ett spel avslutas när antingen en spelare har tre av sina markörer i rad eller då spelbrädet är fullt. I sistnämda fallet blir spelet oavgjort om ingen spelare har tre markörer i rad. Ditt program ska kunna spela på bräden av storlek $k \in \{3, 5, 7\}$. Notera att tre markörers i rad räcker även på $5 \times 5$ och $7 \times 7$ bräden för detta projekt.

## Uppgift 1: Simulera tre i rad

Placering av markeringar är naturligtvis mycket viktigt och avgör spelet. Det finns olika sätt att implementera spelstrategier, men vi ska endast med enkel simulering undersöka start och positionsövertaget spelare 1 har över spelare 2 och 3 om hen sätter första markören i mitten. Vi antar att alla andra drag slumpas ut.

- **Scenario 1:** Alla drag slumpas ut.
- **Scenario 2:** Spelare 1 sätter första markören i mitten. Alla andra drag slumpas ut.



Figur 1: Ett spel av tre i rad på ett $3 \times 3$ bräde med två spelare. Spelare 1 har 'o' som markör.

Du ska skriva ett program som simulerar upprepade partier av tre i rad och undersöker start och positionsövertaget under (den väldigt förenklade) modellen om slumpmässiga placeringar.

- Har spelare 1 en fördel av att börja?
- Blir fördelen större om första markören placeras i mitten?
- Blir fördelen större eller mindre med större brädstorlek?

Skriv ett program som kan utföra upprepade experiment med olika brädstorlekar $k \in \{3, 5, 7\}$ och Scenario 1 eller 2. Användaren ska kunna välja vilket Scenario som ska spelas och hur många partier som ska spelas. Utdata ska ges som ett textmeddelande samt ett stapeldiagram som visar utfallet av experimentet (se fig. 2).

## Exempelkörning

En körning av ditt program ska se ut ungefär så här:

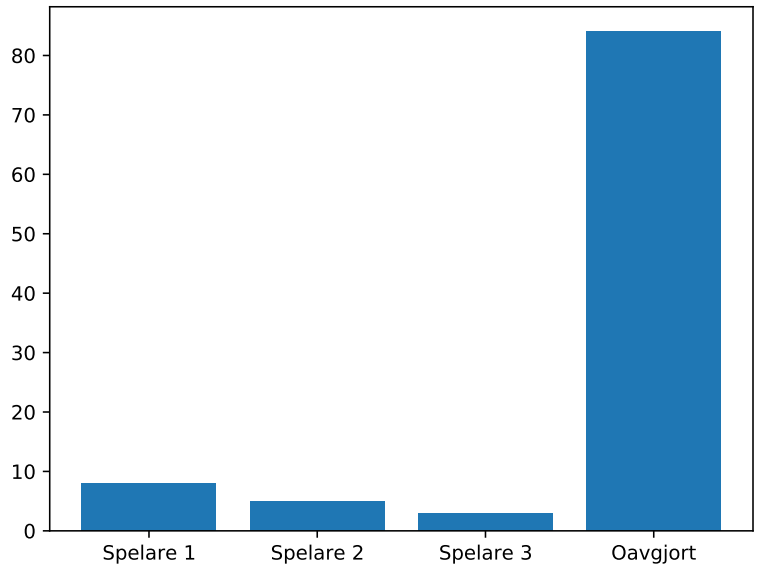
```
How big should the board be (3/5/7)? 5
Which scenario do you want to play (1/2)? 1
How many games do you want to simulate? 100

Player 1 wins: 8
Player 2 wins: 5
Player 3 wins: 3
Draw: 84

A figure that displays the result has been saved in the file "result.pdf"
```

## Krav uppgift 1

- Spelreglerna ska vara korrekt implementerade och visualiseringen samt interaktionen med spelarna ska fungera smidigt.
- Programmet ska inte krascha för felaktig indata från användaren.
- En strategi ska vara generellt implementerad, med hjälp av en funktion eller metod. Samma strategi får inte vara kodad två gånger.




Figur 2: Visualisering av ett experiments utfall med hjälp av `matplotlib`. Obs: påhittade värden.

- Det ska finnas funktionalitet för att testa om det blivit tre i rad givet att en ny markör har blivid placerad. Du får inte fördefiniera alla giltiga vinnande kombinationer av koordinater och testa om det finns i en uppslagstabell.
- Programmet ska använda `matplotlib` för att visualisera resultatet av programmet och spara resultatet i en PDF-fil. Programmet ska inte presentera ett fönster med figuren, utan måste generera en PDF-fil.
- Det får inte finnas diagnostiska utskrifter från varje parti, bara en sammanfattning av resultaten, som i exempeldialogen ovan.

## Uppgift 2: Tredimensionellt tre i rad

Vi ska nu undersöka en tredimensionell variant. I denna uppgift ska vi endast titta på $5 \times 5 \times 5$ brädet (se fig. 3 för hur ett $3 \times 3 \times 3$ kan se ut). Vad som skiljer denna tredimensionella variant från tvådimensionella brädet är att:

- Man kan få tre i rad horisontellt, diagonalt, vertikalt det vill säga åt alla möjliga räta linjer som kan formas tredimensionellt av markörerna.
- Markörerna faller ned till lägsta planet ej innehållandes markör (som kan ses i fig. 3). Man kan därför inte välja den vertikala positionen på markören i kuben.

På samma sätt som i det tvådimensionella fallet är spelet slut när kuben är full.

Vi ska undersöka samma vetenskapliga frågor:

- Har spelare 1 en fördel av att börja?
- Blir fördelen större om första markören placeras i mitten?

Figur 3: Ett spel av tredimensionellt tre i rad på ett $3 \times 3 \times 3$ bräde med två spelare.

## Exempelkörning

En körning av ditt program ska se ut ungefär så här (där `x`, `y` och `z` ska vara tal):

```
Which scenario do you want to play? 1
How many games do you want to simulate? 100

Player 1 wins: x
Player 2 wins: y
Draw: z

A figure that displays the result has been saved in the file "result.pdf".
```

## Tips

Det finns många sätt att konstruera kod för dessa uppgifter. Ett tankesätt: Har man en väl designad kod för uppgift 1 kan man tänka på varje 'lager' i det tredimensionella fallet som ett separat från uppgift 1 och utöka funktionaliteten att testa tre i rad mellan lager. På så sätt kan man återanvända kod, vilket alltid är bra!

## Krav uppgift 2

Korrekt implementation av reglerna för $5 \times 5 \times 5$ bräde vilket uppfyller samma krav som i "Krav för uppgift 1" ovan. För högre betyg måste man även uppfylla de betygskriterier som beskrivs i de generella instruktionerna i *INDU—Individuell Uppgift i Programmering*.