Ricochet Robots

Programmeringskonkurrence

Forfattere: Asger Juul Brunshøj, Anders Roy Christiansen, Signe Colding-Jørgensen,
Andreas Halkjær From, Marcus Skov Hansen, Lasse Kokholm, Gandalf Saxe og Hjalte Wedel Vildhøj.

1 Introduktion

Formålet med denne konkurrence er at designe og implementere et program, der kan spille brætspillet *Ricochet Robots*. De indsendte programmer bliver sat til at dyste mod hinanden, og gruppen, der har lavet det bedste program vinder selvfølgelig en præmie!

Man kan deltage i konkurrencen i grupper bestående af op til fire studerende, og deltagelse tæller som en bestået obligatorisk afleveringsopgave for alle gruppemedlemmer. Der skal ikke skrives nogen rapport. Flere praktiske informationer findes sidst i dette dokument. Bemærk dog følgende vigtige datoer:

7. april kl. 12.30-14.30 Kom-og-kod-workshop med Signe Colding-Jørgensen, Andreas H. From og kage. Her vil det være muligt at få hjælp og diskutere sin løsning med andre. Sted: Bygning 324/040.

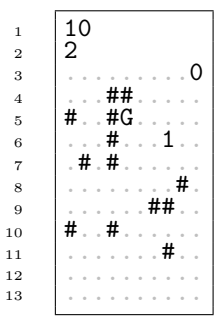
6. maj kl. 20.00: Deadline for at uploade sit program til CodeJudge.

12. maj kl. 10.30: Præmieoverrækkelse. Resultatet af konkurrencen præsenteres og vinderne annonceres. Sted: Bygning 116/81.

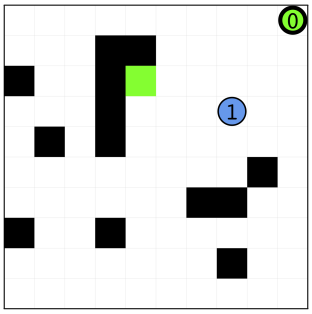
1.1 Kort om brætspillet

I Ricochet Robots gælder det om at bevæge brikker (kaldet *robotter*) rundt på en forudbestemt bane, så en bestemt robot (fx den grønne) ender på en bestemt position (Se Figur 1). En robot kan på et træk bevæge sig enten op, ned, til venstre eller til højre, hvormed den fortsætter i den valgte retning indtil den støder på en mur, en anden robot eller banens kant. Det gælder om at få robotten til at ende på målfeltet med færrest mulige samlede træk, altså inklusive træk med eventuelle andre robotter. For at opnå en løsning med få træk, er det meget ofte en fordel (og nogle gange også nødvendigt) at flytte andre robotter end den robot, der skal hen på målet.

Figur 1: Brætspillet Ricochet Robots



(a) Brættet `2-simpleboard.txt`. De grå prikker er mellemrum.



(b) Visualisering af `2-simpleboard.txt`.

Løsning 1

```
OD
OL
OU
OR
OU
OL
OU
```

Løsning 2

```
1R
0D
0L
```

(c) To gyldige løsninger af henholdsvis længde 7 og 3.

Figur 2: Eksempel på en bane af størrelse $n = 10$ med $r = 2$ robotter. De sorte felter er vægge og det grønne felt er målfeltet for robot 0. Den optimale løsning til denne bane har længde 3.

2 Konkurrencen

I konkurrencen vil vi arbejde med en forsimplet version af brætspillet, der bevarer spillets unikke udfordringer. Denne version spilles på et kvadratisk gitter (kaldet banen) bestående af $n \times n$ felter, hvor $2 \leq n \leq 1000$. I banen findes r robotter, hvor $1 \leq r \leq 10$. Robotterne er nummeret $0, \ldots, r-1$, og det gælder altid om at få robot 0 hen på det unikke målfelt. Robotterne bevæger sig som i brætspillet og stopper således først når de rammer en væg, en anden robot eller banens kant. **Bemærk, at robot 0 skal stå stille på målet for at banen er løst!**

Figur 2 viser et eksempel på en bane med $n = 10$ og $r = 2$ samt to mulige løsninger. I de næste to sektioner beskrives det formelle input- og outputformat samt kravene til jeres program.

2.1 Inputformat

Inputtet til jeres program er en bane som vist i Figur 2a. Den første linje angiver banestørrelsen n som et heltal mellem 2 og 1000. Den anden linje angiver antallet af robotter r som et heltal mellem 1 og 10. De efterfølgende n linjer angiver banen. Hver af disse linjer består af n symboler, der repræsenterer indholdet på denne position i banen. De lovlige symboler i en linje er:

`#` : Angiver at positionen er en væg.

(mellemrum) : Angiver at positionen er tom.

`G` : Angiver at positionen er målpositionen for robot 0.

`0, 1, ...,` $(r-1)$: Angiver at robotten med dette tal er på denne position.

Det garanteres, at symbolet `G` og robotterne `0, 1,` $\ldots (r-1)$ optræder på præcis én position i banen. Det garanteres desuden at banen kan løses.

2.2 Outputformat

En gyldig løsning til banen er en sekvens af træk, der bringer robot 0 hen og i hvile på målpositionen. Hvert træk udskrives som en linje bestående af præcis to symboler: Et heltal i, hvor $0 \leq i \leq r-1$, efterfulgt (uden mellemrum) af et af symbolerne `R` (højre), `L` (venstre), `U` (op) eller `D` (ned). Fx angiver `0R` at robot 0 flyttes mod højre. Et eksempel på to gyldige løsninger er vist i Figur 2c.

2.3 Krav til jeres program

Jeres program skal indlæse en bane fra konsollen (std.in), beregne en gyldig løsning og udskrive denne løsning til konsollen (std.out). Det er ikke et krav at jeres program finder den korteste løsning, men jo kortere løsningen er, desto flere points får jeres program.

3 Sådan kommer du igang – og anden praktisk information

Start med at hente filen `rr.zip`, der ligger på CampusNet under fildeling. Filen indeholder en skabelon i hhv. Java og C++ med forskellige brugbare metoder til fx indlæsning af banefiler m.m. Prøv at uploade filen `Ricochet.java` til CodeJudge – så burde den klare den første test.

Det er ikke et krav at tage udgangspunkt i disse skabeloner – I må også meget gerne skrive jeres løsning i andre sprog, der er understøttede på CodeJudge.

Filen `rr.zip` indeholder desuden en række forskellige baner, som I kan teste jeres løsning på. Alle banefilnavne starter med et tal, der angiver, hvor mange robotter, der er i banen.

Herunder følger yderligere relevant information.

Piazza Brug Piazza til alle spørgsmål, der ikke er besvaret her.

Gruppedannelse Husk at danne gruppe på CodeJudge. Maksimum er fire studerende per gruppe. Aflevering foregår ved at uploade sin løsning til CodeJudge.

CodeJudge Konkurrencen foregår på CodeJudge og er opdelt i to delopgaver: *Obligatorisk* og *Konkurrence*. Den obligatoriske del består af forholdsvis nemme baner med en enkelt robot. Start med at lave en løsning til denne del. Konkurrencedelen indeholder større og mere udfordrende baner med op til 10 robotter.

Den obligatoriske del Uploades en løsning, der består alle tests (5 synlige og 5 skjulte) i den obligatoriske del, tæller dette som en bestået obligatorisk afleveringsopgave for alle medlemmer af gruppen.

Konkurrencedelen Efter deadline tager vi den nyeste version af jeres program i konkurrencedelen og lader det dyste mod de andre programmer på en ukendt samling af baner.

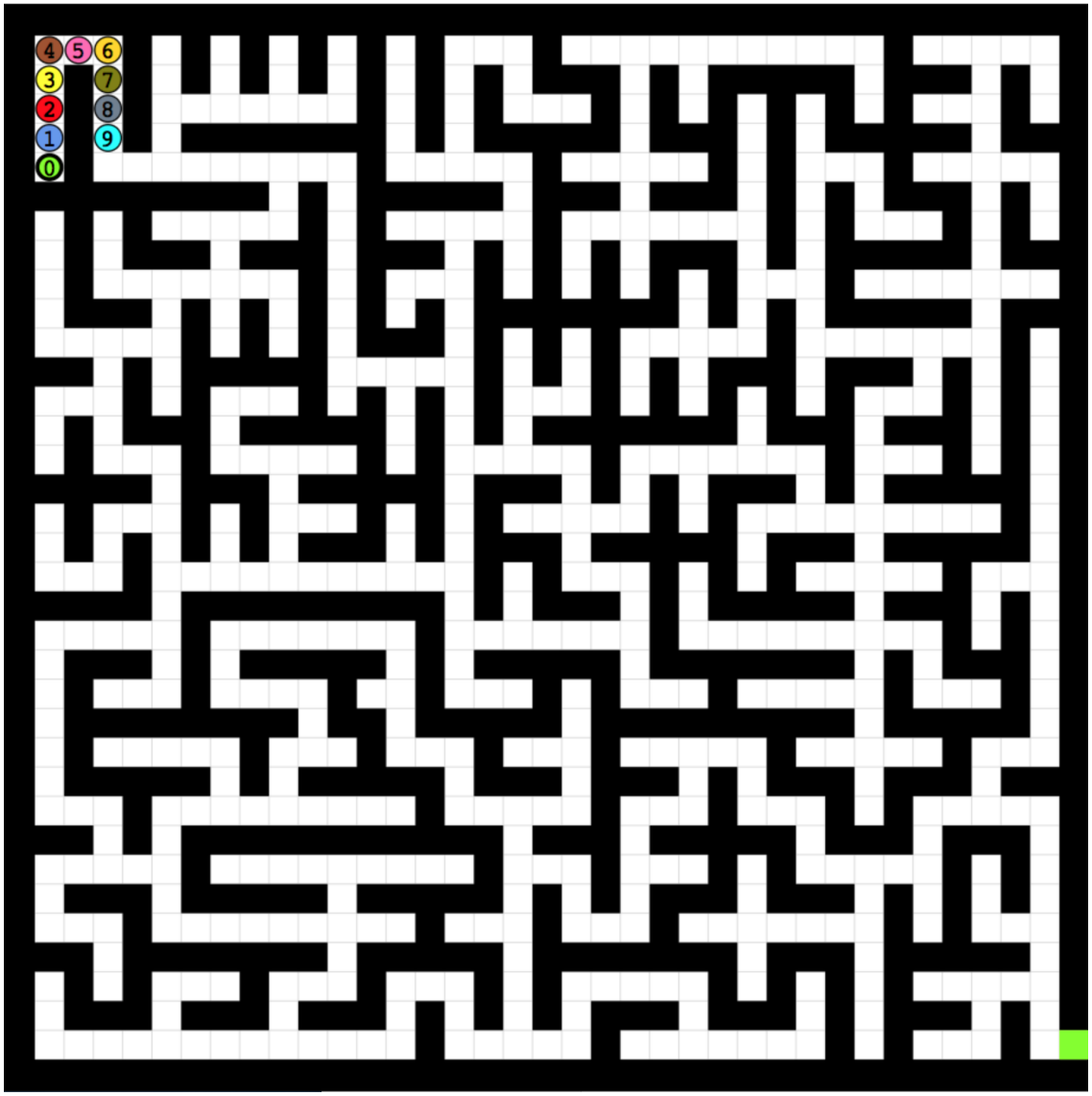
Tids- og hukommelsesgrænser På CodeJudge bliver jeres program tildelt T sekunders CPU-tid og S MB hukommelse. I kan læse værdierne T og S som de første to kommandolinjeargumenter til jeres program. Som udgangspunkt kan I regne med at $T = 5$ sekunder og $S = 500$MB, men når den endelige konkurrence afvikles efter deadline, bliver jeres program tildelt en del flere resourcer.

Pointtildeling For hver bane i konkurrencedelen som jeres program løser får det $100\frac{E}{U}$ points, hvor U er længden af jeres løsning, og E er et estimat af den bedst mulige løsning for banen.

Scoreboard På Scoreboardet på CodeJudge kan man løbende følge med i, hvordan ens løsning klarer sig i forhold til andre gruppers løsninger. Grupperne er sorteret efter det totale antal points deres løsning har opnået på konkurrencebanerne. I tilfælde af pointlighed er den bedste gruppe den med det korteste tidsforbrug. Bemærk, at vi til den endelige konkurrence også tester jeres program på nye og ukendte baner, så derfor er det ikke sikkert, at den endelige vinder bliver den gruppe, der ligger øverst på Scoreboardet.

Webapp På hjemmesiden `https://hwv.dk/rr/` findes en app, hvor man kan importere/eksportere baner og løsninger – og selv prøve at spille banen vha. tastaturet eller mus/touch. Man kan også komme direkte til denne app ved at klikke på visualiseringen af en bane i CodeJudge. Appen er kun testet i Chrome og Safari, men burde også virke i andre browsere og på tablets og touchenheder. Bugs kan mailes til Hjalte på `hwv@hwv.dk`.

Nogle gode råd Start med at lave en løsning, der virker for helt små baner, hvor der kun er en robot. Overvej, hvordan du kan bruge algoritmiske redskaber fra kurset som fx bredde-først-søgning og hashing til at søge efter en løsning.



Figur 3: En af de lidt mere udfordrende baner... (Klik på den!)