Desková hra ŠTREKY

1 Motivace

Úkolem semestrální práce bude vytvořit autonomního hráče pro deskovou hru ŠTREKY. V této hře se tvoří barevné cestičky skládáním kartiček obsahujících fragmenty cest. Karty jsou čtvercové nebo obdélníkové a jsou na nich vyznačeny barevné (pravoúhlé) cesty. Karty se umisťují na hrací desku, nově umisťovaná karta se musí dotýkat alespoň jednou hranou (jedním čtvercem). Karty lze rotovat. Cílem hry je pokládat karty tak, aby se prodlužovaly barevné komponenty souvislosti. Za každou prodlouženou komponentu souvislosti získává hráč body podle jejího obsahu. Hráč bude podroben nejprve hře proti Brutovi, následně pak v Turnaji proti všem odevzdaným programům.

1.1 Pravidla hry

- Hraje se na omezené desce o známých rozměrech, hra je pro dva hráče
- Hráči mají k dispozici několik karet, karty jsou čtvercové nebo obdélníkové a obsahují barevná pole
- Na začátku hry mají hráči stejný počet stejných karet (každý hráč má svoji sadu)
- Průběh hry (hráči se střídají):
  - První tah: hráč na tahu umístí jednu ze svých karet kamkoliv na hrací desku (karta musí být celá na hrací desce)
  - Další tahy: hráč na tahu umístí jednu ze svých dosud nepoložených karet tak, aby byla celá na hrací desce, a zároveň se musí alespoň jednou svoji buňkou dotýkat nějaké již položené karty
  - Je nutné uvažovat i rotace karet (definice rotace viz zadání HW08)
  - Hráč na tahu musí umístit nějakou kartu pokud je to možné
  - Každou kartu lze umístit pouze jednou (hráč si musí pamatovat, které karty již umístil)
  - Je možné, že hráč dostane více karet stejného typu (tvaru a rozdělení barev) — v tom případě může kartu použít maximálně tolikrát, kolik jich obdržel na začátky hry
  - Pokud hráč žádnou kartu nemůže umístit (pokud je hrací deska zaplněná nebo již hráč žádnou kartu nemá), hráč se vzdává tahu
  - Pokud ani jeden z hráčů nemůže umístit žádnou kartu, hra končí
- Bodování:
  - Pokud při položení karty dojde ke zvětšení již existujících barevných oblastí (oblast je souvislá množina buněk stejné barvy), dostane hráč za každou buňku každé zvětšené souvislé oblasti jeden bod
  - Body umístěné v jednotlivých tazích se sčítají
  - Hráč s větším počtem bodů je vítězem hry

1.2 Odevzdání, bodování a turnaj

- Implementovaného hráče odevzdávejte do Brute v souboru `player.py` do úlohy SEM
- Je možné odevzdat pouze jeden soubor (`player.py`), samozřejmě je možné jej odevzdávat opakovaně
- Hráč bude nejprve ohodnocen na náhodných hrách proti Brute hráčovi (očekáváme cca vyšší stovky tahů, vyhodnocení může trvat až 15 minut)
- Při hře na Brute není nutné vyhrávat, ale pouze hrát dle pravidel a dodržet časové limity; Pokud hráč neudělá ani jednu chybu, bude bodové ohodnocení 5 bodů, jinak bude bodové ohodnocení 0 bodů

- V druhé polovině prosince 2024 začne turnaj, kde budou hrát všichni hráči proti sobě v několika hrách
- Hráči, kteří udělají aspoň jednu chybu, budou z aktuálního kola turnaje vyloučeni
- Funkční hráči budou ohodnoceni součtem bodů, které získají v jednotlivých hrách
- Hráči odevzdaní v Brute budou automaticky převedeni do turnaje, turnaj se bude pouštět cca jednou denně
- Poslední týden výuky bude (přesné datum bude včas zveřejněno) finální turnaj
- Body z tohoto turnaje budou přičteny k bodům ze semestrální práce na základě pořadí úspěšných (bezchybných hráčů) seřazených sestupně dle celkového počtu bodů takto:

Finální pořadí v turnaji	**Body**
prvních 10%	+5
dalších 10%	+4
dalších 10%	+3
dalších 10%	+2
dalších 10%	+1

1.3 Implementační detaily

- Student dostane třídu `BasePlayer` (soubor `base.py`), kterou rozšíří v souboru `player.py` (viz balíček na stránce semestrální práce)
- Hráče implementujte do třídy `Player`, do souboru `player.py` (pouze tento soubor se odevzdává na Brute)
- Výpočet jednoho tahu (metoda `play()`) a konstruktor hráče (`__init__()`) může trvat max. 1 sekundu.
- Pokud hráč překročí některý z výše uvedených timeoutů, je označen za chybného
- Konstruktor hráče obdrží velikost hracího pole, a seznam karet, které má uživatel k dispozici
- Metoda `play(newCardOnDesk)` realizuje jeden tah hráče.
  - Pokud protihráč v předchozím tahu položil nějakou kartu, je `newCardOnDesk = [row, col, cardMatrix ]` , kde (row, col) je pozice horního levého rohu karty v hrací desce, cardMatrix je 2D zápis hrací karty
  - Pokud protihráč nemohl hrát, je `newCardOnDesk=[]`
  - Metoda vrací pole `[row, col, cardMatrix ]`, které udává, že hráč chce zapsat kartu na pozici (row,col) na hrací desku (udává se pozice horního levého rohu karty)
  - Pokud hráč žádnou kartu nemůže umístit, je výsledek `[]`
  - Hráči si sami udržují seznam karet, které použili a které obdrželi od protihráče
  - Při hře se (z Bruta) volá pouze metoda `play`, a při vytvoření hráčů jsou nastaveny počáteční hodnoty proměnných (viz dále), jinak Brute do proměnných hráče nikterak nezasahuje
- Význam proměnných, které jsou k dispozici:
  - `self.boardRows`, int, je počet řádků hrací desky (nastaveno Brutem)
  - `self.boardCols`, int, je počet sloupců hrací desky (nastaveno Brutem)
  - `self.cardsOnDesk`, list, je pole karet na hrací desce (udržuje si hráč pokud chce)
  - `self.cardsAtHand`, list, je pole karet, které dostal uživatel na začátku hry (nastavuje Brute)
  - `self.userLogin`, str, je jméno uživatele (vyplňuje Brute)
  - `self.playerName`, str, je jméno, které si hráč sám zvolí (vyplňuje hráč ve svém konstruktoru)
  - `self.tournament`, bool, je `True` v Turnaji, a `False` na Brutovi (nastavuje Brute)
    - Touto proměnnou můžete zapnout/vypnout chytré strategie (nejsou potřeba pro Brute, ale jen pro turnaj)

```
if self.tournament:
   #chytra strategie pro umistovani karet
else:
   #jednoducha strategie pro Brute, neni treba vyhravat
```

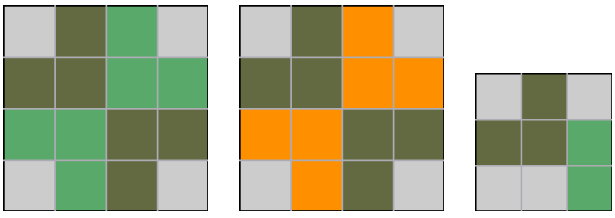
- Karty, které jsou na hrací desce, jsou uloženy v `self.cardsOnDesk = [ cardPos1, cardPos2, ... ]`
- Jedna karta je reprezentována jako `cardPos = [ row, col, cardMatrix ]`, kde row,col je souřadnice levého horního rohu karty na hrací desce, a `cardMatrix` je 2D matice reprezentující kartu. Tato 2D matice obsahuje pouze hodnoty 0 (prázdné pole bez barvy), nebo barevná pole (hodnoty 1 až 4).
- Počet řádků karty je `len(cardMatrix)`, počet sloupců karty je `len(cardMatrix[0])`
- Hráč si sám zvolí vnitřní způsob reprezentace hrací desky a informaci o umístěných kartách; není třeba používat `self.cardsOnDesk`, ale je to doporučené
- Předpokládáme číslování od 0, přičemž horní levý roh hrací desky má souřadnice (0,0)
- Brute odchytává veškerý standardní výstup a neposkytuje jej uživateli, tj. při hře proti Brute a v turnaji budou odfiltrovány výsledky `print()` apod.
- Není garantován přístup na disk (tj. pokud si chcete ukládat nějaké výpočty během hry, používejte radši paměť (pole, dict) než soubory); toto omezení plyne z konfigurace výpočetního gridu, kde se bude spouštět turnaj
- Hráče lze napsat s využitím standardních funkcí Pythonu, není třeba importovat nějaké knihovny (byť to nezakazujeme), ale v takovém případě kontaktujte přednášejícího (ověří, že vámi požadované knihovny budou k dizpozici v turnajovém režimu)
- Nepoužívejte v programech diakritiku a jiné znaky než v základní verzi ASCII a to ani v komentářích

1.4 Příklad karet k dispozici

- Hráč má karty k dispozici v proměnné `self.cardsAtHand`, což je pole 2D matic reprezentujících karty
- Dejme tomu, že tato proměnná obsahuje

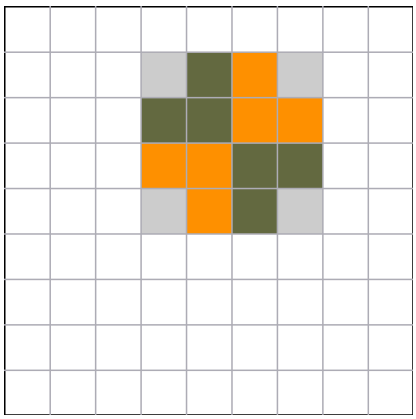
```
self.cardsAtHand = [ [[0,1,2,0],[1,1,2,2],[2,2,1,1],[0,2,1,0]],
                     [[0,1,4,0],[1,1,4,4],[4,4,1,1],[0,4,1,0]],
                     [[0,1,0],[1,1,2],[0,0,2]]
                   ]
```

- tj. jsou k dispozici tři karty (dvě o rozměru 4×4 a poslední o rozměru 3×3)
- Pokud bychom přiřadili barvy takto: 1=tmavě hnědá, 2=zelená a 4=oranžová, pak lze tyto karty interpretovat:



1.5 Příklad umístění karty na hrací desku

- Uživatel se rozhodne umístit kartu `self.cardsAtHand[1]` předchozího příkladu na pozici (1,3) (řádek, sloupec).
- Výsledek metody `play` bude `return [1, 3, [[0,1,4,0],[1,1,4,4],[4,4,1,1],[0,4,1,0]] ]`
- Hráč, který tuto kartu použil, ji nesmí znova umístit (ani její rotaci) — tj. je třeba si poznamenat, že karta na 1. pozici již byla použita.



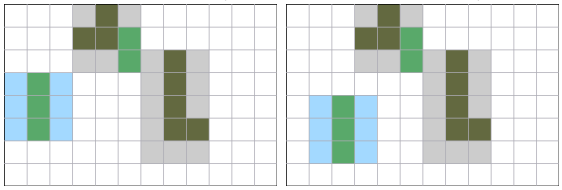
- Protihráč obdrží tuto hodnotu v proměnné `newCardOnDesk`
- Pokud chce hráč umístit orotovanou verzi nějaké karty, musí jaké výsledek metody `play` vrátit její rotovanou verzi
- Například pokud chceme umístit kartu č. 2 (`self.cardsAtHand[2]` z předchozího příkladu) orotovanou (counterclockwise o 90 stupňů), na pozici 3,4, bude výsledek `play()`:

```
return [3,4, [ [0,2,2],[1,1,0],[0,1,0] ] ]
```

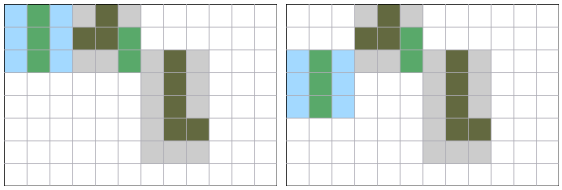
1.6 Povolené umístění karet

Kartu je třeba dávat na hrací desku tak, aby se dotýkala nějaké již položené karty hranou, tj. aby nově položená karta a již existující karta sdílely alespoň jednu buňku hrací desky (bez ohledu na jejich barvu). Příklady umístění karty (modře zvýrazněné) do hrací desky (již umístěné karty jsou šedivé):

Nesprávné (nedotýká se hranou)



Správné



1.7 Testování offline

Na stránce semestrální práce (`https://cw.fel.cvut.cz/b241/courses/b3b33alp/cviceni/t09`) si stáhněte balíček, který obsahuje prázdného hráče (`player.py`) a základní třídu (`base.py`). Souboru `player.py` lze spustit (bud' v editoru/IDLE/VSCode apod.), nebo v příkazové řádce:

```
>python3 player.py
```

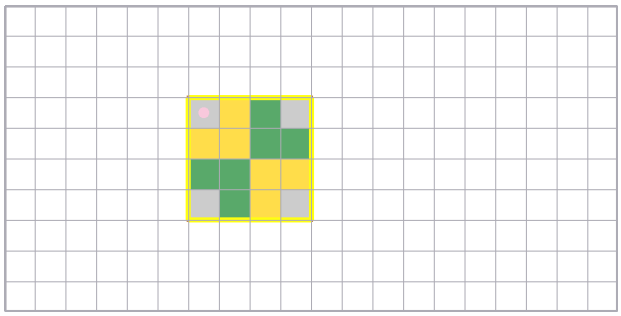
Program je napsán tak, aby "hrál sám proti sobě". Záznam hry je k dispozici jako seznam PNG souborů. Výchozí soubor `player.py` pouze náhodně umist'uje karty a nekontroluje validnost tahů.

1.8 Tipy

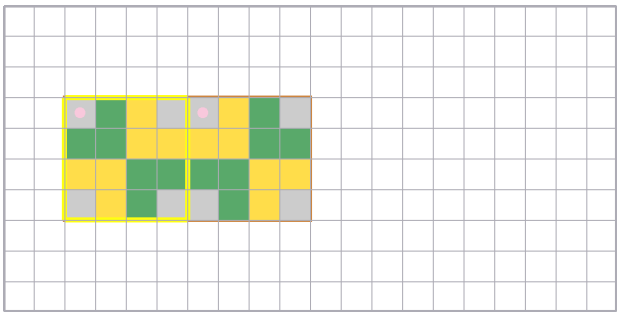
- Doporučujeme použít programy z úlohy HW08
- Nejprve implementujte hráče, který hraje dle pravidel (bez ohledu na skóre), teprve pak přemýšlejte jak odhadovat další tahy tak, abyste získali co největší skóre
- Jelikož mají oba hráči na začátku stejnou sadu karet, lze přesně dopočítat, s jakými kartami může protihráč hrát
- Pokud chcete měřit rychlost výpočtu, použijte `time.time()` z modulu `time` (bylo na přednáškách)
- Pokud narazíte na časové limity, zvažte, které operace je nutné provádět v metodě `play` a které (pokud takové jsou) lze předpočítat (např. v konstruktoru nebo při prvním tahu).

- Sledujte jak hrajou ostatní hráči (na stránkách turnaje budou detailní logy), zkuste proti nim najít strategii
- Důrazně doporučujeme nepoužívat globální proměnné
- Doporučujeme pojmenovávat proměnné smysluplně (aby vyjadřovaly, co obsahují)
- Pokud chcete používat svoje vlastní třídy, definujte je též v souboru `player.py`

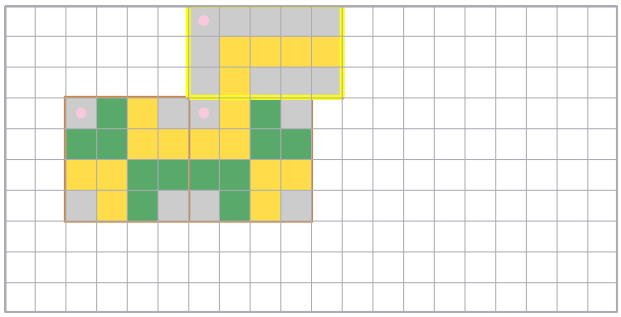
1.9 Příklad hry



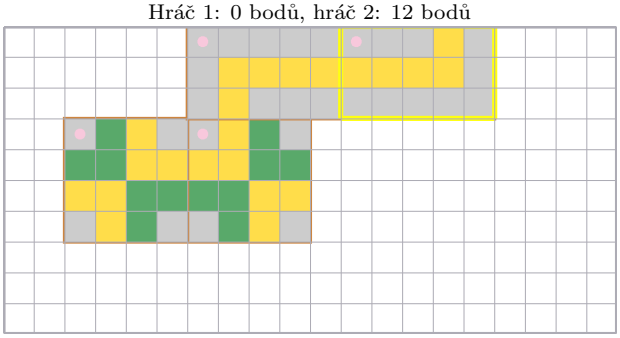
Tah 1: Hráč 1 pokládá libovolně na hrací desku
Hráč 1: 0 bodů, hráč 2: 0 bodů



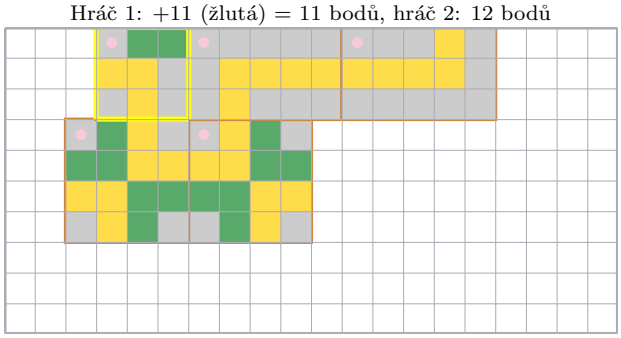
Tah 2: Hráč 2 pokládá tak, aby se dotýkal předchozích karet
Došlo k prodloužení oranžových cest (+6 bodů) a tmavých (+6 bodů)
Hráč 1: 0 bodů, hráč 2: 12 bodů



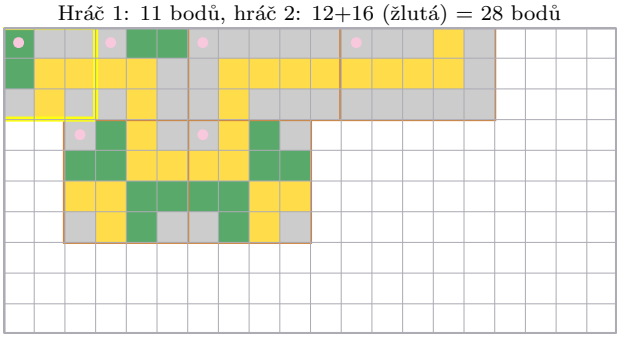
Tah 3: Hráč 1 pokládá žlutě orámovanou kartu
Hráč 1: +11 (žlutá) = 11 bodů, hráč 2: 12 bodů



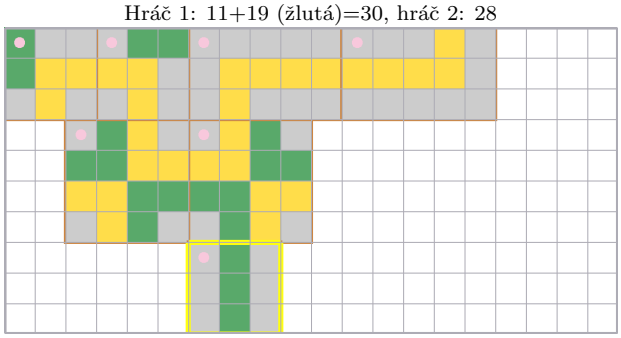
Tah 4: Hráč 2 pokládá žlutě orámovanou kartu
Hráč 1: 11 bodů, hráč 2: 12+16 (žlutá) = 28 bodů



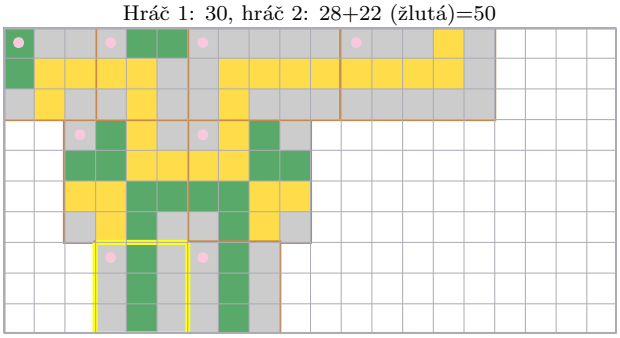
Tah 5: Hráč 1 pokládá žlutě orámovanou kartu
Hráč 1: 11+19 (žlutá)=30, hráč 2: 28



Tah 6:Hráč 2 pokládá žlutě orámovanou kartu
Hráč 1: 30, hráč 2: 28+22 (žlutá)=50



Tah 7: Hráč 1 pokládá žlutě orámovanou kartu
Hráč 1: 30+9 (zelená)=39, hráč 2: 50



Tah 8: Hráč 2 pokládá žlutě orámovanou kartu
Hráč 1: 39, hráč 2: 50+12 (zelená)=62