Dokumentace k zápočtovému programu - Hra Života

Daniel Bartušek

January 9, 2021

1 Stručné zadání

Tento program implementuje algoritmus Hry života od Johna Conwaye. Hra života simuluje život buněk ve čtvercové matici, ve které jednotlivé buňky jsou buď živé nebo mrtvé. Tento stav se mění po kolech, na základě stavu sousedních buněk. Program dále Hru života rozšiřuje o zajímavé interaktivní "módy". Kromě klasické simulace program nabízí typ hry "Doplnění do symetrie", kdy hráč může vstoupit do hry a pozměnit až 3 políčka tak, aby pole bylo symetrické. Další typ se jmenuje Sázka na život. Hráč si podle počátečního pole vsadí na buňku, která podle něj zaznamená nejvíce přežitých kol.

2 Přesné zadání

Při spuštěnní programu je od hráče požadováno specifikovat parametry hry. Hráč volí velikost hracího pole v rozmezí 2-26. Poté hráč volí jeden z typů hry: klasická simulace, doplnění do symetrie, sázka na život. Klasická simulace reprezentuje standartní průběh hry. Pole se mění na základě algoritmu Hry života, a hráč nemá možnost do hry vstoupit.

Doplnění do symetrie dává hráči možnost v jakémkoliv momentu vstoupit do hry, a změnit stav až tří políček. Symetrie se posuzuje podle osy y. Po zadání se naposledy vypíše stav pole, a vypíšou se pole, která nejsou symetrická.

Sázka na život vyžaduje po hráči, aby si podle stavu počátečního pole zvolil buňku, která podle jeho úsudku přežije nejvíce kol. Při smrti buňky se počet neresetuje, pouze pozastavuje. Hráč vyhrává, pokud na konci hry je právě jeho buňka ta s nejvíce přežitými koly.

Maximální počet kol je pro každý mód nastaven na 10. Pro každé kolo se vypíše přehledná tabulka ukazující stávající pole. Mezi koly je nastavená uživatelsky přivětivá sekundová prodleva. Tato prodleva přihodně testuje reflexy při typu hry Doplnění do symetrie, jelikož dává hráči pouze sekundu na správný úsudek, zda jde pole udělat symetrické.

Po hráči je dále požadováno zadání způsobu tvorby pole. Možnosti výběru jsou: náhodné pole, vlastní pole a předdefinované útvary.

Vlastní pole dovoluje uživateli používat 'O' nebo '1' pro zápis živé buňky, a '-' nebo '0' pro zápis mrtvé buňky. První varianta je povolená, jelikož tyto dva znaky reprezentují stav buňky při tisknutí jednotlivých kol. Zápis pomocí 1/0 je povolen pro konvenčnost. Pokud uživatel 2x odenteruje v konzoli, program mu podá informaci, kolik ještě políček zbývá zaplnit. Dokud nejsou

zadané všechny buňky, program nepokračuje.

Předdefinované tvary jsou speciálně vybrané pro jejich vlastnosti. Existují stovky zajímavých útvarů pro Hru života, tento program nabízí tři z nich: Pentomino, Fumarole a Boatmaker. Za účelem přehledné ukázky útvarů je tento typ hry povolen jen v případě, že hráč zadal velikost pole alespoň 20.

Každý ze vstupů hráče je ošetřen tak, aby byl vždy korektní. V případě nekorektního vstupu je hráčí poskytnuta nápověda, co zadal špatně.

3 Popis algoritmu Hra života

Čtvercová síť hry života je tvořena mrtvými a živými buňkami, stav buňky v dalším kole je vyhodnocen těmito pravidly:

- Žijící buňka s jediným žijícím sousedem do příštího kola nepřežije.
- Žijící buňka se dvěma nebo třemi sousedy přežívá do dalšího kola přežívá do dalšího kola.
- Mrtvá buňka se třemi žijícími sousedy obživne,
- Mrtvá či žijící buńka obklopená 4 žijícími sousedy nemůže žit kvůli přepopulovanému okolí.

4 Program

Program je rozdělen na dvě části. První z částí je samotná třída HraciPlocha. Třída obsahuje vstupní parametry této třídy jsou velikost pole, zvolený typ hry, způsob tvorby počátečního pole a případná volba předdefinovaného útvaru pro počáteční pole. Druhá část představuje interaktvní zadání parametrů hry, a samotný průběh hry.

4.1 Hrací plocha

Hrací plocha je tvořena listem listů (sloupců). Náhodně tvořené pole postpně plní jednotlivé sloupce znaky '-' nebo 'O', a výsledný sloupec vloží do prázdného listu představující hrací pole.

V případě tvorby vlastního pole jsou sloupce v poli předem nadefinovány. Předpokládáme, že uživatel bude chtít své pole definovat spíše přes řádky, než přes sloupce, jelikož konzole dovoluje psát vstup po řádcích. Pro čtení znaků je použitá funkce sys.stdin.read(1) z modulu sys. Znaky ze vstupu jsou tedy rovnovážně ukládány napříč sloupci, a posouvá se ukazatel pozice ve sloupci, ne ukazatel samotného sloupce, který se má plnit.

Tvorba počátečního pole je možná i s použitím předdefinovaných útvarů. V takovém případě je vytvořeno pole mrtvých buněk '-'. Stav vybraných buněk je pak změněn relativně ke středové souřadnici pole.

4.2 Vyhodnocení nového pole

Mezi koly je potřeba vyhodnotit nastávající rozložení pole. To probíhá po sloupcích. Na začátku se vytvoří nový list naplněný mrtvými buňkami. Pro každou buňku v daném sloupci z původního pole se vyhodnotí stav jejich sousedů (relativně k souřadnicím dané buňky). Pokud je sousední buňka živá, zvyšuje se proměnná obsahující počet živých sousedů. V novém sloupci pak může buňka obživnout na základě počtu jejich žijících sousedů, a pravidel definovaných v předchozí sekci "Popis algoritmu Hra života".

Pokud si uživatel vybral typ hry "Sázka na život" tak se souběžně zvyšuje počet přežitých kol buňky v pomocném poli (V případě, že buňka má v novém sloupci stav 'O'. Funkce novyTah() toto nové pole dosadí do aktuálního stavu pole.

4.3 Tiskárna pole

Tiskárna rozlišuje dva typy pole - průběžný stav buněk, a konečný počet přežitých kol buněk. Tisk probíhá po řádcích. Kromě samotného pole se dále tiskne řádek s označením sloupců (a-z) a na začátku každého řádku se tiskne číselné označení řádku. Vše je pro přehlednost ošetřeno o odsazení.

4.4 Vyhodnocení nejdéle žijících buněk

Pro vyhodnocení se prochází jednotlivě všechny buňky, a hledá se maximální hodnota přežitých kol. Pokud se hodnota ve hráčem zvoleném poli shoduje s touto hodnotou, uživatel vyhrál.

4.5 Symetrie

Hráč má v rámci typu hry "Doplnění do symetrie" možnost změnit stav tří buněk. Symetrie se kontroluje podle osy y, a provádí se po řádcích. Pokud má pole lichý počet sloupců, prostřední sloupec se nekontroluje.

4.6 Změna tří polí

Pro zpracování vstupu obsahující 3 hráčem zadané buňky je vstup rozdělen do listu těchto buněk. První část buňky je písmeno označující sloupec. Toto písmeno je převedeno na číselné označení sloupce. Zbylá část je číslo označující řádek. Pole se řádkují od jedničky, takže je od vstupního označení odečtena jednička. Po zpracování je v buňce s danými souřadnicemi otočen stav.

4.7 Parametry hry

Další složka programu je interaktivní nastavení parametrů hry. Každý vstup je ve While loopu, který se rozbije až při při zadání požadovaného tvaru vstupu. Samotný průběh programu je také ve while loopu, který se rozbije v momentě, kdy počet kol dojde k 10.

5 Alternativní programová řešení

Vzhledem k uniformitě obsahu buněk, totiž znaků, by bylo možné na tvorbu pole použít rychlejší a přehlednější numpy modul. Dále pro grafické zobrazení pole by bylo příhodné použít nějaký

grafický modul, jako turtle.

6 Reprezentace vstupních a výstupních dat

Vstup je po uživateli požadován postupně v konzoli. Uživatel musí většině případů pouze zadat písmeno nebo číslo jedné z představených možností. Vše je doprovázené patřičnými instrukcemi. Výstupem jsou tisknuté tabulky a doprovodné věty popisující situaci.

7 Průběh prací

7.1 Algoritmus hry

První jsem se rozhodl naprogramovat samotné vyhodnocení pole podle pravidel algoritmu. SLožitější na tom bylo správně ošetřit sousedy buněk tak, aby se program nesnažil zjišťovat stav neexistujících buněk venku z pole. Dále představovat si pole v prostoru pouze podle listů, a bez pomocného tisku, nebylo jednoduché.

Další problematická část pro mě bylo správné nastavení odsazení, aby tisk buněk a jejich počtů byl vždy správně zarovnán s označením sloupců. Původně jsem chtěl odsazení určovat pomocí řádu buněk s nejvýše přežitými koly, ale nepřišel jsem na to, jak tuto hodnotu vložit do zápisu ” print(”%2d” %radek, end = ” ”)”. Takže jsem nakonec usoudil, že uživatel nebude chtít sledovat více než 100 kol hry, a nastavil odsazení na dva řády.

Poslední pro mne složitou části byl vstup hráče do hry pro případ ”Doplnění do symetrie”. Program neustále vyhodnocuje nová pole, takže bylo třeba nějakým inputem od uživatele tento proces zastavit. Nakonec jsem to vyřešil zachycením vyjímky. Jsem si vědom toho, že to není správné využití vyjímek jako takových.

Ukázka:

```
while kolo<=10:
    life.tiskarna(life.stav)
    kolo+=1
    t0 = time.time()
    try:
        print("Pokud si prejete vstoupit tento tah, stisknete ctrl+c")
        time.sleep(1)
    except KeyboardInterrupt:
        zmenaTed = True
        break
    life.novyTah()
```