



155YOBP

SEMESTRÁLNÍ PROJEKT

HRA ŽIVOTA

Skupina B:

Marek Hoffmann

Monika Křížová



OBSAH PREZENTACE

1 | Představení hry života



2 | Vysvětlení postupu a kódu



3 | Řešení problémů

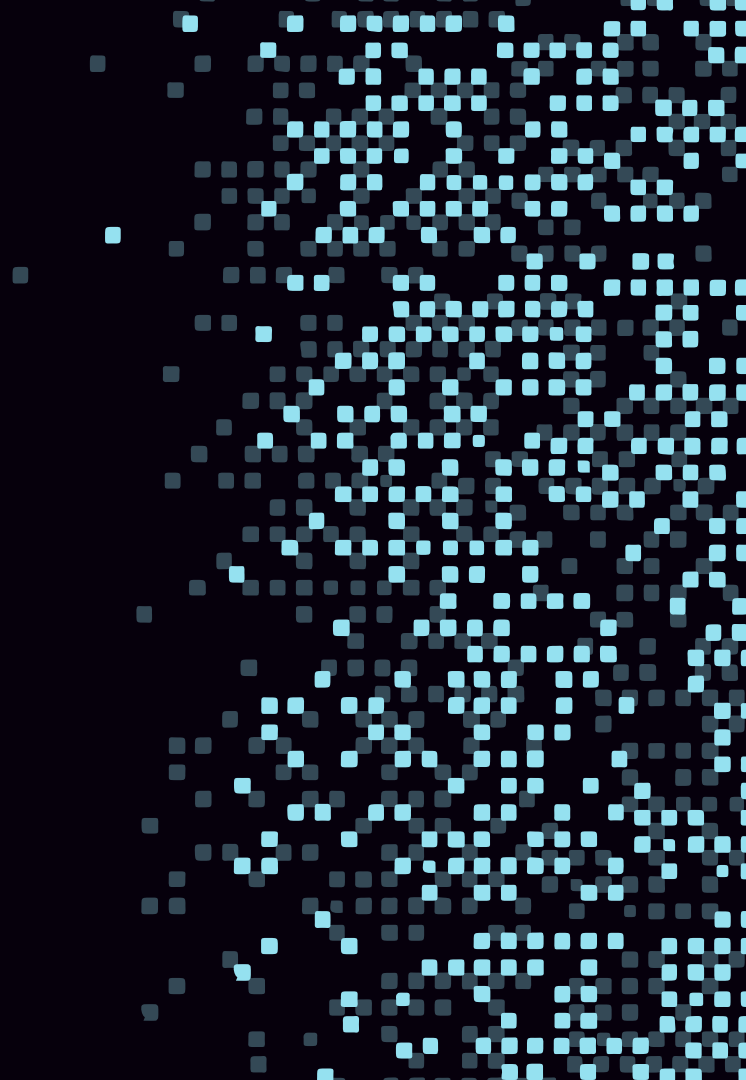


4 | Dotazy a diskuze



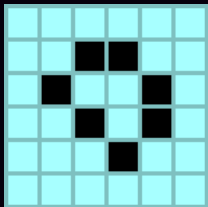
HRA ŽIVOTA

- Dvojměrný celulární automat
- **4 základní pravidla:**
 - Každá živá buňka s méně než dvěma živými sousedy zemře.
 - Každá živá buňka se dvěma nebo třemi živými sousedy zůstává žít.
 - Každá živá buňka s více než třemi živými sousedy zemře.
 - Každá mrtvá buňka s právě třemi živými sousedy oživne.

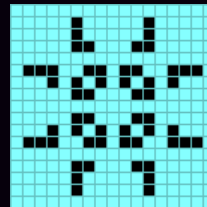


ZÁKLADNÍ TVARY HRY

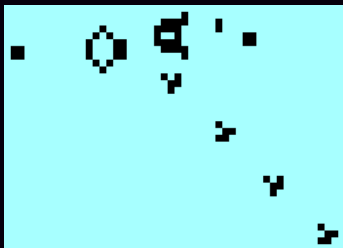
zátiší



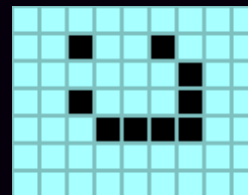
oscilátory



děla



posunující se
tvary



OBSAH PREZENTACE

1 | Představení hry života



2 | Vysvětlení postupu a kódu



3 | Řešení problémů



4 | Dotazy a diskuze



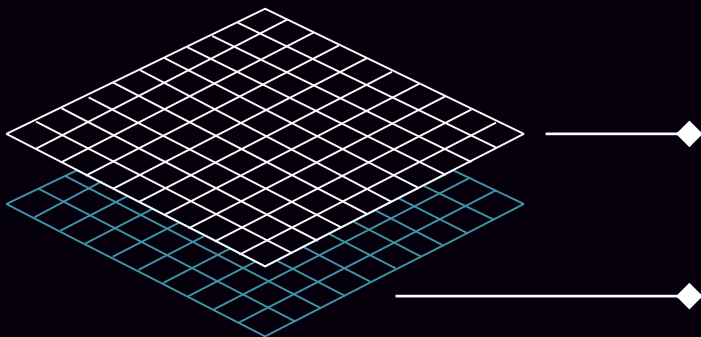
PROGRAMOVACÍ JAZYK A SOFTWARE



random



HERNÍ MŘÍŽKY



aktivní mřížka

`self.mrizky[0][radek][sloupec]`

neaktivní mřížka

`self.mrizky[1][radek][sloupec]`



živá buňka

`self.mrizky[1][radek][sloupec] = 1`



mrtvá buňka

`self.mrizky[1][radek][sloupec] = 0`



STRUKTURA KÓDU

spust(self):

1. self.zpracuj_akce()

ovládání hry uživatelem (z klávesnice)
s – pozastavení hry
r – náhodné přestavení mřížky
q – ukončení hry

2. self.oprav_generaci()

vyšetří současnou generaci a připraví generaci budoucí

- pro každou buňku analyzuje počet sousedů
- určí stav buňky v příští generaci a uloží jej do neaktivní mřížky
- vymění neaktivní mřížku za aktivní

4. self.fps()

ovlivňuje rychlost vykreslování mřížky

3. self.kresli_mrzkou

vyčistí obrazovku a následně každé buňce mřížky vloží barvu podle jejího stavu (živá/mrtvá)

while true



OBSAH PREZENTACE

1 | Představení hry života



2 | Vysvětlení postupu a kódu



3 | Řešení problémů



4 | Dotazy a diskuze



ŘEŠENÍ PROBLÉMŮ

Ošetření správného vracení hodnot okrajových buněk

```
def vrat_stav_bunky(self, r, s):  
    if r < 0 or s < 0 or r > self.pocet_radku - 1 or s > self.pocet_sloupcu - 1:  
        hodnota_bunky = 0  
    else:  
        hodnota_bunky = self.mrizky[self.aktivni_mrizka][r][s]  
    return hodnota_bunky
```



ŘEŠENÍ PROBLÉMŮ

Chyba při vykreslování jednotlivých buněk

PŮVODNÍ ŘEŠENÍ

```
def init_mrizka(self):  
  
    self.pocet_sloupcu =  
    int(SIRKA/VELIKOST_BUNKY)  
    self.pocet_radku = int(VYSKA/  
    VELIKOST_BUNKY)  
  
    self.mrizky = ([[0] * self.pocet_radku] *  
    self.pocet_sloupcu,  
                    [[0] * self.pocet_radku]*  
    self.pocet_sloupcu)  
    self.aktivni_mrizka = 0;
```

NOVÉ ŘEŠENÍ

```
def init_mrizka(self):  
    def vytvor_mrizku():  
  
        radky = []  
        for cislo_radku in range(self.pocet_radku):  
            temp = []  
            for cislo_sloupce in range(self.pocet_sloupcu):  
                temp.append(0)  
            radky.append(temp)  
        return radky  
  
    self.mrizky.append(vytvor_mrizku())  
    self.mrizky.append(vytvor_mrizku())
```



OBSAH PREZENTACE

1 | Představení hry života



2 | Vysvětlení postupu a kódu

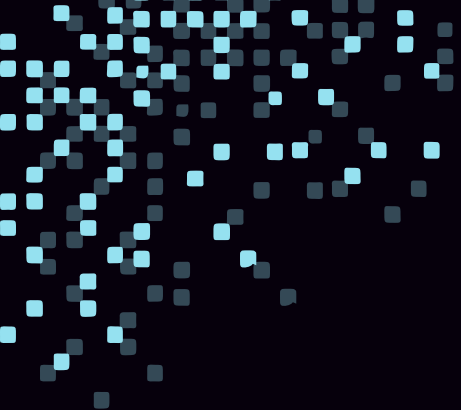


3 | Řešení problémů



4 | Dotazy a diskuze





DĚKUJEME ZA POZORNOST

<https://github.com/ctu-yobp/2020-b>

