Hra p^11 jako lepší 2048

Zápočtový program k předmětu NMIN112, LS 2023

Autor: Matyáš Maroušek

Datum: 16. 8. 2023

Obsah

Uživatelská dokumentace

Úvod programu

Program je koncipován jako ´klasická hra 2048´ s několika rozšiřujícími prvky, jako například možnost změny základu mocnin na jiná prvočísla, ukládání nejvyššího dosaženého skóre a celková ´barevnost´ pozadí i čtverečků s nenulovými hodnotami, takže se hráč může během hraní kochat.

Tha map of links & hlavní funkce/obrazovky

MENU: Po spuštění se načte obrazovka se světlým pozadím a herním menu, které obsahuje název hry – „p^11“, resp. obecněji jako „p^n“, odkazy v podobě tlačítek „PLAY“, „PRIMES“ a „EXIT“ a nápisy s dosavadně nejvyšším dosaženým skóre „Highscore“ a základ mocnin, se kterými by se po spuštění hry jako takové hrálo, „Base“.

EXIT: Aktivováním odkazu – kliknutím myší na tlačítko „EXIT“ ukončíte program stejně, jako byste kliknuli na ´červený křížek´ vpravo nahoře tohoto okna. Oběma těmito akcemi samozřejmě nenávratně smažete i herní pokrok. Tlačítko „EXIT“ naleznete pouze v herním menu.

PRIMES: Tlačítkem „PRIMES“ v herním menu se přesunete na obrazovku s vysvětlením principu postupu ve hře – dosažením skóre alespoň p^12-p, kde p je nejvyšší odemknutý základ, se odemkne základ o hodnotě dalšího prvočísla (nejvyšší odemknutelný základ je 7). Základ změníte kniknutím na tlačítko s jinou číselnou hodnotou. I na této obrazovce se ukazuje vaše highscore i vybraný základ mocnin. Jediným odkazem na jinou obrazovku je tlačítko „MENU“, kterým se dostanete zpátky do herního menu.

PLAY: Tlačítkem „PLAY“ se spustí hlavní část hry. Přejdete na obrazovku se světlým pozadím, tabulkou 4x4 čtverců, nápisy a hodnotami „Score“ a „Highscore“ a tlačítkem „END THIS TRY“. Hru ovládáte knikáním na šipky (doleva, doprava, nahoru, dolu) – na šipkami určenou stranu se, kam až je možné, posunou všechny nenulové čtverce – pokud by při tomto pohybu v tomto směru ´narazily´ na čtverec o stejné hodnotě, spojí se s ním v jeden se základ-násobnou hodnotou. K takové kombinaci může v jedné linii dojít nejvýše dvakrát pro čtveřici hodnot. Skóre a highscore se změní podle aktuálních hodnot v tabulce, resp. v závislosti na doposud nejvyšším skóre. Při každém kliknutí na šipku, které uvnitř hry vyvolá změnu v tabulce, se zároveň na místě libovolného po pohybu volného, nulového čtverce vygeneruje čtverec s hodnotou vybraného základu. Aktuální pokus hry končí ve chvíli, kdy se při libovolném pohybu neuvolní čtverec pro vygenerování nového nenulového, nebo kniknete a tlačítko „END THIS TRY“ nebo na ´křížek´ v pravém horním rohu. Tlačítko „PLAY“ naleznete pouze v herním menu.

END THIS TRY: Tlačítkem „END THIS TRY“ okamžitě ukončíte rozehraný pokus a na okně se zobrazí obrazovka s nápisem „Game Over!“, pokud jste během ukončeného pokusu překročili nebo dosáhly hodnoty dříve nejvyššího skóre, uvidíte i nápis „New Record!“, dále nápisy s příslušnými hodnotami „Score:“ a „Highscore:“ a na spodní části okna tlačítko „MENU“. Netřeba dodávat, že s tlačítkem "END THIS TRY“ se setkáte pouze na obrazovce herního pokusu.

Před spuštěním samotného programu je nutné mít nainstalovaný python, pravděpodobně libovolnou ze současných posledních verzí, aktivované knihovny/moduly Pygame, NumPy, random a sys, ačkoli je možné relativně snadno program přepsat tak, aby poslední tři knihovny/moduly nebyly zapotřebí. Dále je nutné, aby součástí složky, kde se vyskytuje soubor kódu pod názvem „better 2048“, byly obrázky „endthistry_obr“, „exit_obr“, „five_obr“, „menu_obr“, „play_obr“, „primes_obr“, „seven_obr“, „three_obr“, „two_obr“, které jsou využívány jako tlačítka v programu.

Programátorská dokumentace

Basics

Program se v základu sestává ze šesti částí;

1) Import slovníků a modulů a inicializace pygame,
2) čtyři slovníky přiřazující přirozeným číslům souřadnice v RGB modelu barev, import a inicializace knihoven,
3) třídu „butOn“ využívající knihovnu „pygame“ pro používání tlačítek,
4) funkci „gen_svetle_pozadi()“ generující souřadnice v RGB modelu barev souhlasné s nějakou relativně světlou barvou pomocí knihovny „random“,
5) třídu „Game_p_Asterisk_asterisk_n“ obsahující veškeré funkce s cykly, uvnitř kterých se program odehrává
6) spuštění funkce „menu()“ třídy „Game_p_Asterisk_asterisk_n“ po spuštění programu

Knihovny a moduly

Knihovnu „numpy“ využíváme pouze pro snazší manipulaci s herní tabulkou, knihovna „pygame“ je nejdůležitější knihovnou programu a mimo jiné nám slouží k vytvoření okna programu, načtení obrázků, využití je jako třídu „Rect“ a veškeré manipulace s oknem programu, knihovnu „random“ využíváme pouze pro generaci náhodného světlého pozadí a modul „sys“, abychom mohli vynutit ukončení programu bez chybových hlášení.

Slovníky barev

Barvy ve slovnících pro jednotlivé mocniny byly vybrány takřka náhodně s podmínkami, aby mezi ´sousedními´ nebyl drastický rozdíl a ani jedna nenabývala odstínu šedé. Alternativně se nabízí vygenerovat tyto barvy interaktivně hráčem za pomocí několika jednoduchých funkcí.

Class buttOn

Třída „buttOn“ je koncipována jako primitivní tlačítko, pro jehož vytvoření stačí výběr nahraného obrázku a souřadnice levého horního rohu tohoto obrázku, délku a šířku buttOn přejímá z délky a šířky obrázku. Jeho součástí jsou dvě podstatné metody; zobrazení buttOnu („make_button(screen)“) a detekce kliknutí myší na oblast uvnitř nebo na okraj buttOnu.(„buTton_clicked()“).

Metoda „**make_button(screen)**“ zobrazí obrázek spojený s buttOnem jako obdélník s levým horním rohem na vložených souřadnicích v okně „screen“, ačkoli pro načtení se zobrazení tohoto buttOnu na okně „screen“ je ještě potřeba metody „update()“ nebo „flip()“ provedené na „screen“ („screen“, ačkoli je pouhým obecným argumentem libovolného okna, uvažujeme jako odkaz na nějaké předem jednoznačně dané okno).

Metoda „**buton_clicked()**“ vytváří dvouprvkový tuple souřadnicí šipky myši v okně programu a kombinací booleanů „akce“ a „self.clicked“ zamezuje vícenásobný output True/False při jediném intervalu (ne)kliknutí.

Generating random light color

Funkce „**gen_svetle_pozadi()**“ využívá dvou funkcí modulu „random“; „randrange()“ a „sample()“ pro generaci pseudo-náhodného trojčlenného tuple-u – souřadnice relativně světlé barvy v RGB modelu barev.

The main class

Třída „**Game_p_Asterisk_asterisk_n**“ obsahuje veškerou herní logiku včetně všech cyklů, ve kterých se hráč může vyskytnout. Nejdůležitějšími atributy jsou „self.num“ – počet čtverců na jedné hraně samotné hry (defaultně = 4, ale s trochou práce lze dát hráči možnost výběru této ´šířky´, zda bude chtít hrát s polem 5x5 či jiným – a od toho odvodit velikost okna, nutné skóre pro odemknutí dalších základů atd.), „self.base“ – prvotní základ mocnin ve hře při spuštění (defaultně nastaveno na = 2, protože základní formou je ´klasická hra 2048 s polem 4x4´), „self.score“ – aktuální skóre, „self.high_score“ – doposud nejvyšší dosažené skóre, a „self.okno“ – okno, ve kterém se celý program odehrává a ukazuje uživateli.

__init__(): mimo již zmíněných a souvislých parametrů a atributů se zde ze složky, kde se nachází program, načítají obrázky, následně se z nich vytváří butOny včetně jejich umístění v okně a vytváříme i tři různé fonty.

make_it_Start(): je snadno nahraditelná funkce a slouží pouze k přesměrování od spuštění programu k hernímu menu.

menU(): je funkce, která vykreslí úvodní obrazovku – herní menu obsahující nápisy název hry, highscore („Highscore:“), aktuálně používanou bázi („Base:“) a tři tlačítka, u kterých díky while-cyklu můžeme využívat jejich metodu buTton_clicked() – kliknutím na libovolné z nich se ukončí cyklus pro menu a začne jiný, resp. se ukončí celý program.

Tlačítkem „PLAY“ se aktivuje funkce „plaY()“ a započne hra

Tlačítkem „PRIMES“ se aktivuje funkce „priMes()“ a přejde se na obrazovku výběru základů mocnin

A tlačítkem „EXIT“ se ukončí celý program

Funkce **priMes()** vykreslí obrazovku s popisem cíle, respektive postupu hry, a zobrazí 1 až 4 číselná tlačítka v závislosti na výši highscore, kterými lze změnit základ mocnin uvnitř hry, a hodnoty aktuálně nastaveného základu a highscore. Kliknutím na libovolné číselných tlačítek se změní hodnota self.base na příslušnou hodnotu a opětovaně se spustí funkce priMes(). Posledním tlačítko má nápis „MENU“, kterým se z tohoto while-cyklu také vystoupí, ovšem ne do nového cyklu funkce priMes(), nýbrž funkce menU().

Funkce **exit()** slouží univerzálně k ukončení běhu programu funkcemi pg.quit() a sys.exit(). K její aktivaci dochází kdykoli v průběhu běhu programu, když stisknete červený „QUIT“ křížek okna, a když kliknete na tlačítko „EXIT“, které se nachází pouze v herním menu.

Nejpodstatnější částí programu je herní cyklus uvnitř funkce **plaY()**, která nejdříve vytvoří nulovou čtvercovou matici self.herni_pole o straně self.num, funkcí Add_number na libovolné nulové místo přidá číslo self.base (náš základ) a aktualizuje skóre funkcí update_Score().

```
        self.herni_pole = np.zeros((self.num, self.num))
        self.Add_number()
        self.update_Score()
```

Před cyklem se nachází ještě příkazy pro uložení náhodné světlé barvy vybrané pro pozadí, vybarvení pozadí, přidání nápisů a hodnot Score a Highscore a zobrazení tlačítka pro okamžité ukončení hry („END THIS TRY“). Uvnitř cyklu se funkcí Make_a_board() zobrazí tabulka matice self.herni_pole s příslušnými hodnotami a zbarvením a pro každý děj (event.) uvnitř hry kontrolujeme, zda jde o kliknutí na ´quiting křížek´ okna, na „END THIS TRY“ buttOn nebo libovolnou, kterými se hra obsluhuje – šipkou se spustí funkce move(směr_šipky). Po těchto dějích se kontroluje funkcí is_game_over(), zda jsou splněny podmínky pro konec hry. V případě, že hra není u konce, kontrolujeme, zda při tomto eventu došlo ke změně v herním poli – pokud ne, nic se nemění a probíhá

reakce na další zaznamenaný event., pokud ano, spouští se funkce Add_number(), update_Score() a přepisují se nápisy hodnot self.score a self.high_score díky zakrytí obdélníkem barvy odpovídající pozadí a znovunapsáním hodnoty.

Funkce **Make_a_board()**: prochází číslo po čísle v self.herni_pole a na odpovídajících místech v self.okno vzhledem k souřadnici čísla se vytvoří čtverec s barvou z knihovny barev odpovídající základu self.base a hodnotě čísla společně s nápisem tohoto čísla. Např.:

```
if self.base == 2:
   pg.draw.rect(self.okno,
                barvicky_2[hod_ctverce],
                pg.Rect(souradnice_x, souradnice_y, self.hrana_ctverce,
                        self.hrana_ctverce))
```

Pár řádků kódu je věnováno správnému výběru velikosti fontu nápisu hodnoty čísla, aby nápis nevylézal za hranice příslušného čtverce.

Funkce **movE(směr_šipky)**: prochází matici self.herni_pole po řádcích, resp. sloupcích, - záleží na směru posunutí – „line“, v jistých směrech posunutí (doprava a dolu) využívá obrácení tohoto vektoru čísel, aby logika v pomocné vektorové funkci Kombine(line), která je na tento vektor line zavolána – v případě, že v line byla dvojice stejných čísel vedle sebe, čísla se sloučila do jednohoa a vektor line je o jedno místo kratší. Pokud je délka upraveného vektoru menší než self.num, je ve správném směru doplněn o nuly.

```
            line = self.Kombine(line)

            line = line + (self.num - len(line)) * [0]
```

Po případném opětovném obracení vektoru funkce vrací upravený vektor „line“ na původní souřadnice do self.herni_pole.

Pomocná funkce **Kombine(line)**: ignoruje nuly v line a od strany směru „směr_šipky“ po dvojicích spojuje stejná sousední čísla do mocniny čísla self.base o jeden řád vyšší, nové číslo ukládá do pomocného vektoru „outcome“ a za každou spojenou dvojici do pomocného vektoru ukládá nulový prvek. Pokud číslo nemá sobě rovného souseda, je beze změny uloženo do outcome. Nakonec jsou z outcome opět vymazány všechny nuly a funkce vrací vektor outcome.

Pomocná funkce **Add_number()**: vytvoří seznam nulových čtverců v self.herni_pole a do libovolného jednoho z nich vloží namísto nuly číslo self.base.

Pomocná funkce **update_Score()**: nejdříve nastaví self.score na nulu, načež projde celou matici self.herni_pole, odkud z při-tomto-tahu-již-nové matice přičte k self.score hodnotu z každého prvku. V případě, že je self.score větší nebo rovno self.high_score, přepisuje se i hodnota self.high_score.

Funkce **is_game_oveR()**: kontroluje, zda by se při pohybu libovolným směrem herní tabulka, resp. matice self.herni_pole, jakkoli změnila – pokud již nelze vykonat tah, aby se stav hry nějak změnil, nastává konec hry a funkce vrací hodnotu True. V opačném případě konec hry ještě není.

```
    def is_game_oveR(self):
        kopie_pole = self.herni_pole.copy()

        for direction in "LRUD":
            self.movE(direction)

            if (self.herni_pole == kopie_pole).all() == False:
                self.herni_pole = kopie_pole
                return False

        return True
```

Funkce **gAme_over()**: je spuštěna při výsledku True z aktivované funkce is_game_oveR() nebo při kliknutí na tlačítko „END THIS TRY“. Stejně jako plaY(), menU() a priMes() přepíše původní obrazovku pseudo-náhodnou světlou barvou. Přidá nápis „Game Over!“ společně s nápisy „Your score:“, „Highscore:“, odpovídající hodnoty. V případě, že se během ukončené hry přepsalo self.highscore nebo se dosáhla stejná hodnota skóre jako bylo původní highscore, zobrazí se i nápis „New Record!“. Současně je spuštěn while-cyklus, při kterém je zobrazeno tlačítko „MENU“, jehož zmáčknutí self.score nastaví na nulu, spustí funkci menU() a ukončí tento cyklus, a stejně jako vždy lze cyklus ukončit i quitnutím ´červeným křížkem´.

Spuštění programu

Program končí částí, kde se jeho spuštěním vytvoří instance „game“ reprezentující celou třídu Game_p_Asterisk_asterisk_n, na tuto instanci je zavolána metoda make_it_Start(), která spustí funkci metodu menU() třídy Game_p_Asterisk_asterisk_n.

```
if __name__ == "__main__":
    game = Game_p_Asterisk_asterisk_n()
    game.make_it_Start()
```

Díky této formulaci lze program spouštět jako hru a zároveň jej lze importovat jinam, do jiných modulů, aniž by se samotná hra spustila.

Obecné shrnutí programu

Hra je implementována objektově s využitím tříd pro herní logiku a grafické rozhraní. Obsahuje metody pro inicializaci, zpracování vstupu, vykreslování, herní cyklus včetně ukončení hry.

Shrnutí práce z pohledu autora

Na programu jsem pracoval v průběhu čtyř týdnů, kdy jsem studoval jiné programy her typu 2048 a celkově využívání knihovny Pygame v souvislosti s programováním her, zkoušel různé varianty většiny funkcí, resp. metod, a mnoho dalšího. Možná až ironicky byly největšími výzvami práce v tlačítky, respektive jejich implementace, výběr pozadí okna v průběhu programu a počítání souřadnic a velikostí obrázků a fontů textu. Jsem si vědom, že logicky není složité všechna tato čísla

definovat pomocí několika rovnic závislých na rozměrech obrázků a případně uživatelem zvolenou hodnotou čísla self.num, ale protentokrát jsem se rozhodl takto a více interaktivní variantu programu možná vytvořím v budoucích letech.

Častokrát jsem během programování zjistil, že jsem došel k funkci, resp. metodě nebo algoritmu, které již nějaký jiný uživatel Internetu vytvořil celé roky přede mnou, i když třeba jen trochu jinak. Což na mě jako amatérského programátora působí zároveň nemile, protože v takových případech se nejednalo o něco, co by ´na světě už nebylo´, ačkoli se takové zjištění dalo čekat při vytváření tak známé hry, ale zároveň jako motivace k tomu v budoucnu již začínat s originálnějšími nápady.