Gymnázium, Praha 6, Arabská 14

Obor programování, vyučující Jan Lána

Karetní hra Prší

ročníkový projekt

Jakub Švéda, 1.E

Květen 2021

Prohlašuji, že jsem jediným autorem tohoto projektu, všechny citace jsou řádně označené a všechna použitá literatura a další zdroje jsou v práci uvedené. Tímto dle zákona 121/2000 Sb. (tzv. Autorský zákon) ve znění pozdějších předpisů uděluji bezúplatně škole Gymnázium, Praha 6, Arabská14 oprávnění k výkonu práva na rozmnožování díla (§ 13) a práva na sdělování díla veřejnosti (§ 18) na dobu časově neomezenou a bez omezení územního rozsahu.

V …… dne ……. Podpis: ……

Název práce: Karetní hra Prší

Autor: Jakub Švéda

Anotace: Cílem projektu je vytvořit věrnou kopii karetní hry Prší. Tedy vytvořit grafické zpracování herních karet, dále menu, ze kterého se hra ovládá a v neposlední řadě vytvořit schopnost počítače samotnou hru hrát proti reálným hráčům. To vše za použití programovacího jazyka Java.

Obsah

1. Úvod

Tento dokument popisuje základní pravidla, kterými by se měl uživatel během hry Prší řídit. Rozebírá tak jednotlivá základní pravidla a zároveň řeší rozdíly, při různém nastavení hry. Mimo jiné popisuje implementaci počítačové logiky, design karet, strukturu projektu z hlediska kódu a princip, jakým je tvořeno uživatelské prostředí. Dále uvádí podrobný postup, jakým by se měl uživatel řídit, při spuštění samotné hry.

Zadání

Zadáním tohoto ročníkového projektu bylo vytvořit offline karetní hru Prší, která bude obsahovat několik herních módů:

1. Hráč proti počítači
2. Hráč proti hráči
3. Hráč proti hráči proti počítači
4. Hráč proti počítači proti počítači
5. Hráč proti hráči proti hráči

2. Pravidla hry

2.1. Obecná pravidla

Karetní hra Prší obsahuje celkem 32 herních karet. Všechny se dělí na 4 barvy. Tyto barvy jsou zelená (listy), žlutá (žaludy), modrá (kule) a červená (srdce). Každá barva má celkem osm různých karet. Na začátku hry se karty rozdají po čtyřech každému hráči tak, aby hráči vzájemně nevěděli, jaké mají karty. Následně se položí jedna karta na stůl, kterou všichni vidí. Zbytek balíčku funguje jako lízací balíček. Hráči střídají své tahy podle předem určeného pořadí. Cílem hry je zbavit se jako první všech karet. Kartu může hráč odhodit do hracího balíčku na stole jen, pokud je jeho karta stejné barvy jako karta na stole nebo má stejnou hodnotu. Tento systém má jisté výjimky při hraní speciálních karet, které budou rozebrány vzápětí. Pokud nemůže hráč odehrát ani jednu z jeho karet, musí si jednu kartu líznout.

Sedmička

Pokud někdo hrál sedmičku, následující hráč má dvě možnosti, jak ukončit svůj tah:

a) Pokud má ve svém balíčku také sedmičku, může ji hrát, pak jeho tah končí a pokračuje další hráč. Efekt sedmičky nadále přetrvává.

b) Pokud nemá ve svém balíčku sedmičku, musí si líznout tolik karet, jako je dvojnásobek právě odehraných sedmiček a následně pokračuje další hráč.

Eso

Pokud někdo hrál kartu eso, následující hráč má dvě možnosti, jak ukončit svůj tah:

a) Pokud má ve svém balíčku také eso, může jej odehrát a pokračuje další hráč. Efekt esa nadále přetrvává.

b) Pokud nemá ve svém balíčku eso, musí prohlásit, že stojí a hraje další hráč.

Filek

Tato karta může být hrána v jakémkoli případě bez uvážení barvy nebo stejné hodnoty mimo efekty sedmičky a esa. Hrou této karty si daný hráč vybere barvu, jaká bude ovlivňovat následnou hru. Barva je zvolená do doby, dokud někdo neodehraje kartu zvolené barvy nebo znovu nezmění barvu jiným filkem.

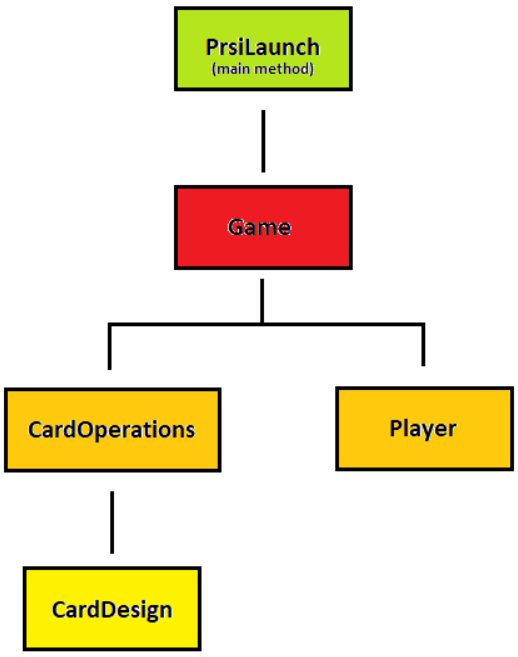
2.2. Modifikovatelná pravidla

Mimo klasická pravidla je v programu možnost některá nastavit. Jedno z nich je, že červená sedmička vrací hráče do hry. Pokud v módech 1 a 2, tedy v případě že hrají dva hráči, některý z hráčů odehrál svou poslední kartu, avšak hráč po něm na řadě má ve svém balíčku červenou sedmičku a podle pravidel ji odehraje, pak potenciální výherce musí líznout 2 karty a hra pokračuje.

3. Strukturace projektu

Projekt byl navržen ve vývojovém prostředí NetBeans IDE (NetBeans, 2021) ve verzi 8.2, které zahrnuje překladač pro programovací jazyk Java.

Celý projekt obsahuje celkem 5 tříd (viz *Obrázek I*). První třída s názvem PrsiLaunch spustí samotnou aplikaci, bez zásahu do kódu hry. Byla vytvořena zejména pro přehlednost kódu.



Obrázek I: *Struktura programu*

Další malou třídou je třída Player, která umožňuje vytvořit objekt Player, který uchovává jméno každého z hráčů.

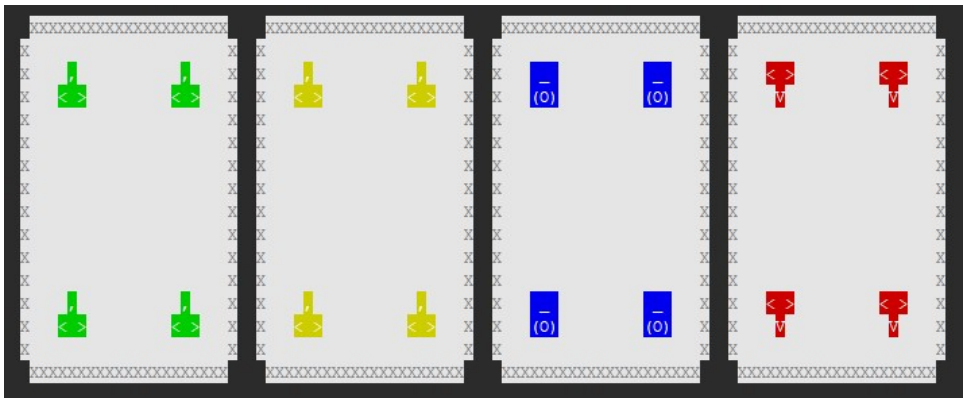
Dále jsou zde hlavní tři metody celé hry, jimiž jsou CardDesign, CardOperations a Game. Třída CardDesign uchovává vzhled karet, třída CardOperations se stará o herní logiku a třída Game se stará o grafické rozhraní pro uživatele. Více v dalších kapitolách.

4. Design karet

Jak již bylo zmíněno, o grafické zpracování karet se stará třída CardDesign. Každá karta je uložena formou jednorozměrného pole datového typu String (řetězce znaků) tak, že každý jednotlivý řádek karty je uložen jako jeden prvek.

Celkem má každá karta šířku 23 znaků a výšku 16 znaků. Kraje karty jsou tvořeny na každém řádku jedním znakem „X“ u každého kraje.

Jejich barvy jsou tvořeny tak, že jsou na bílém pozadí mimo prvky na kartě, jimiž jsou myšleny např. design znaku kule nebo listu (viz *Obrázek II)*. Jejich barva se určuje pomocí předem vytvořených konstant datového typu String standartu ANSI (ANSI escape kód, 2021).



Obrázek II: *Příklad názorné grafiky karet*

Tato třída má svůj konstruktor a obsahuje gettery polí každé z karet. Takto se v průběhu hry nemůže prostřednictvím chyby změnit grafika karet.

5. Logická složka hry

Třída CardOperations tvoří srdce celé hry. Stará se o početní operace hry a stavbu grafického rozhraní, když je třeba při tomto procesu pracovat s kartami. Karty jsou v této chvíli reprezentovány čísly. Na základě jejich identifikačních čísel se kartám přidělují jejich grafické interpretace.

5.1. Míchání karet

Algoritmus pro míchání karet se spustí vždy na začátku hry. Karty jsou zamíchány a zároveň rozdány podle zvoleného herního módu do proměnných typu ArrayList<Integer> každého hráče. Toto ukládání bylo zvoleno místo pole, neboť má volitelnou velikost a je tedy snazší k této délce přistupovat.

Celý proces míchání probíhá tak, že je vytvořeno jednorozměrné pole o délce 32, protože potřebujeme uložit 32 karet. Do tohoto pole se postupně ukládají náhodně generované čísla. Při získání náhodného čísla se vždy ověří, zda vygenerované číslo již není jednou uloženo, duplikace karet je totiž nepřípustná. Když ano, pak se vygeneruje nové náhodné číslo a proces ověření se opakuje. Na konci po zaplnění pole se toto pole vrátí.

S polem se pak pracuje tak, že se jeho prvky rozdají po čtyřech čísel do ArrayListů hráčů, jedna karta do ArrayListu, který funguje jako odhazovací balíček a zbytek náleží ArrayListu lízacího balíčku.

5.2. Knihovna karet

Knihovna karet je jednou z metod ve třídě CardOperations. Umožňuje zobrazit jednotlivé karty, jak vypadají a zároveň k nim přidává popis každé karty. Měla by v případě nejasností jakoukoliv kartu popsat. Ke knihovně karet lze přistoupit z hlavního menu.

5.3. Kontrola pravidel hráče

Hráč si při svém tahu zvolí kartu, kterou by chtěl odhodit. Nyní se načte číslo této karty a program začne kontrolovat, zda je možné touto kartou hrát. Existuje celkem šest základních situací, ve kterých se zvolená karta může vyskytnout.

První kontrolovanou situací je, zda nemůže být odehrána na základě toho, že hráč přede mnou již nemá žádnou kartu a nynější hráč má červenou sedmičku, kterou předchozího hráče může vrátit do hry. V takové situaci může být odehrána právě červená sedmička a žádná jiná karta. V případě neúspěchu opakuje hráč svůj pokus zadání.

Druhá kontrolovaná situace je, pokud je nynější hráč v efektu sedmičky. V této situaci může odehrát jen jinou sedmičku, pokud ji má v balíčku, jinak si musí vzít karty z lízacího balíčku. Tedy podle pravidel dvojnásobek sedmiček za sebou odhozených do odhazovacího balíčku. V případě neúspěchu také opakuje svůj pokus zadání.

Třetí situací, se kterou je možné se setkat, je, když je hráč v efektu esa. Nyní může hrát pouze další eso nebo prohlásit že stojí.

Další situace se přímo táže na zvolenou kartu a to, jestli je zvolená karta filkem (měničem) či nikoliv. V případě že ano, pak nechá hráče, dle jeho uvážení, zvolit barvu, s jakou bude v příštím tahu hráno. Pokud není filkem, pak pokračuje kontrolou další z podmínek.

Pátou situací je, pokud je zvolená barva filkem z předchozích tahů. V tomto případě se ověřuje, zda je karta požadovaná ke hře stejné barvy jako zvolená barva filkem. Pokud ano, hraje ji a pokud to jde, zapamatuje si její schopnost pro další tah/y. Pokud ne, hráč svoje zadání opět opakuje.

Poslední situací je, když ani jedna z předchozích podmínek neplatí a tak, když karta odpovídá stejné barvě nebo alespoň stejné hodnotě jako karta na stole, kartu program hraje.

Po vyhodnocení všech možností vrátí tato kontrolní metoda kladné číslo nebo mínus jedna „-1“. S kladným číslem se dále pracuje v uživatelském prostředí, které je na základě vráceného čísla aktualizováno. Pokud je vrácena hodnota „-1“ znamená to, že zadání bylo chybné. Společně s negativním vyhodnocením se vytvoří i chybová zpráva, která je později v rámci uživatelského prostředí zobrazena.

5.4. Algoritmus počítače

Algoritmus počítače řeší stejné situace jako kontrola pravidel hráče, ale trochu jiným způsobem. Nejdříve za stejných podmínek jako při kontrole hráče otestuje celý balíček a uloží použitelné karty. V tuto chvílí je vybírána nejvhodnější karta k použití již ze seznamu použitelných karet.

První situace kontroluje, jestli není nutné hrát pouze červenou sedmičku v případě, kdy má tato karta vracet druhého hráče zpět do hry.

Druhá situace říká, že pokud je jen jedna použitelná karta, nemá smysl řešit další jiné podmínky a proto je tato karta odehrána.

V ostatních případech je řešení situace, kdy je počítač v efektu esa, tedy dokáže stát nebo odehrát jiné eso. Dále řešení situace, kdy je počítač v efektu sedmičky, dokáže tak lízat podle pravidel nebo hrát nutnou další sedmičku. Když to jde, použije sedmičku, která není červená, aby měl větší šanci vrátit svého protihráče zpět do hry. Nejzajímavější v této sekci jsou situace, kdy se počítač ptá, zda nemá jeho protihráč jen jednu kartu. Pokud ano, snaží se hrát primárně sedmičku. Dále pokud hráč změnil kartu a nyní již má jen jednu kartu, počítač barvu změní na svoji nejpočetnější barvu (Java Program to Find the Largest Number in an Array, 2021). Pokud se barvy shodují se zvolenou kartou protivníka, vybere počítač svoji druhou nejpočetnější barvu.

V ostatních případech hraje počítač karty podle své aktuální nejpočetnější barvy. Jinak bere kartu z lízacího balíčku, neboť nemá jakou kartou hrát.

6. Uživatelské prostředí

O uživatelské prostředí se stará primárně třída Game v kombinaci s CardOperations. Při startu aplikace se zobrazí úvodní text s webovým odkazem na pravidla Prší a následně hlavní menu.

Odtud uživatel vybere mód, dostane se do nastavení, zobrazí knihovnu karet nebo z programu odejde.

V nastavení může zapnout nebo vypnout funkci, kdy červená karta bude vracet do hry. Může se vrátit zpět do hlavního menu.

Při volbě knihovny karet se zobrazí karty podle barev i s jejich popisem. Dále se program automaticky vrátí do hlavního menu.

Při zvolení jednoho z módů se program dotáže na jména všech hráčů mimo jména počítače. Dále zobrazí herní pořadí a v tu chvíli se zeptá, zda může začít první hráč. Po potvrzení karty odkryje. Obdobně se pokračuje dále, tzn. zobrazí se dialog, zda může program pokračovat a hraje se dále. Zbaví-li se některý z hráčů všech svých karet a neplatí pravidlo, že červená vrací do hry, pak je automaticky zvolen výherce a program se vrátí zpět do hlavního menu, které funguje jako metoda s návratovou hodnotou typu boolean. Celý proces hry je uzavřený ve while cyklu (viz *Obrázek III*), kvůli lepší optimalizaci programu.

```
public void launch() {
    startDialog();
    while (mainMenu()) {
    }
}
```

Obrázek III: *Cyklus programu*

Aby si hráči vzájemně neviděli do karet, je nutné je skrýt. Vzhledem k tomu, že zřejmě žádná z konzolí nemá podporu ANSI znaků a zároveň nedokáže jednoduše zajistit mazání konzole v průběhu programu, tak má hra k mazání jiný přístup. Pokaždé, když je nutné smazat předchozí řádky vypsané do konzole, tak program odřádkuje několik prázdných řádků. K tomu aby vše fungovalo, je třeba nastavit v nastavení konzole jen omezenou historii (viz kapitola *Spuštění aplikace*).

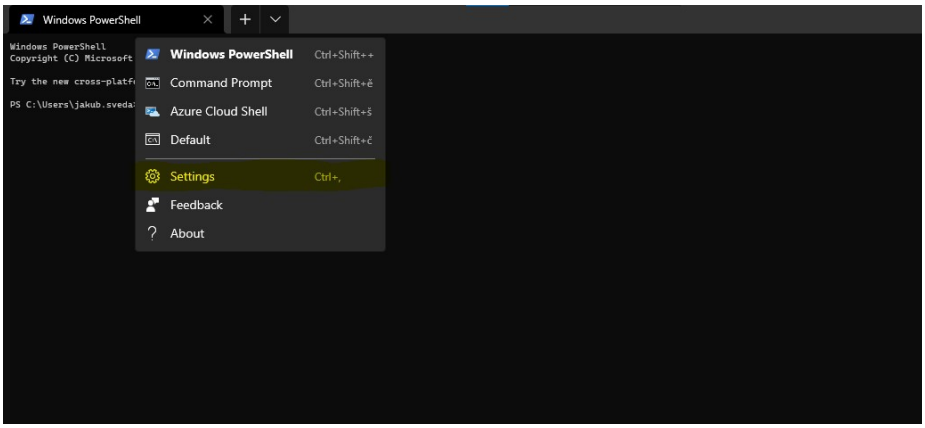
7. Spuštění aplikace

Aby aplikace fungovala v plném rozsahu, je nutné dodržet několik základních nastavení a hlavně hrát v podporované konzoli.

Jako první by se měl uživatel ujistit, zda má operační systém Windows 10 a staženou Javu. Ostatní operační systémy nejsou nativně podporovány, to však neznamená, že program nebude na jiných fungovat. Jen je nutné dodržet některá pravidla.

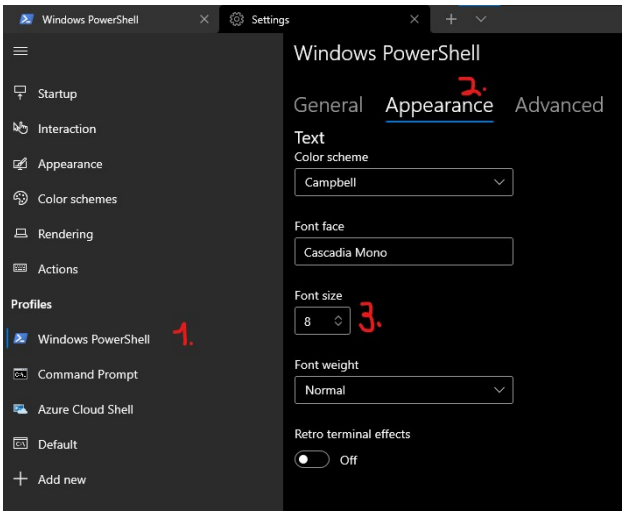
Pro nejlepší herní zážitek je nutno nainstalovat aplikaci Windows Terminal zdarma dostupnou v Microsoft Store (Get Windows Terminal, 2021). Je možné použít i ostatní konzole, ale musí podporovat ANSI escape znaky.

Po instalaci aplikaci spustíme a otevřeme její nastavení (viz *Obrázek IV).*



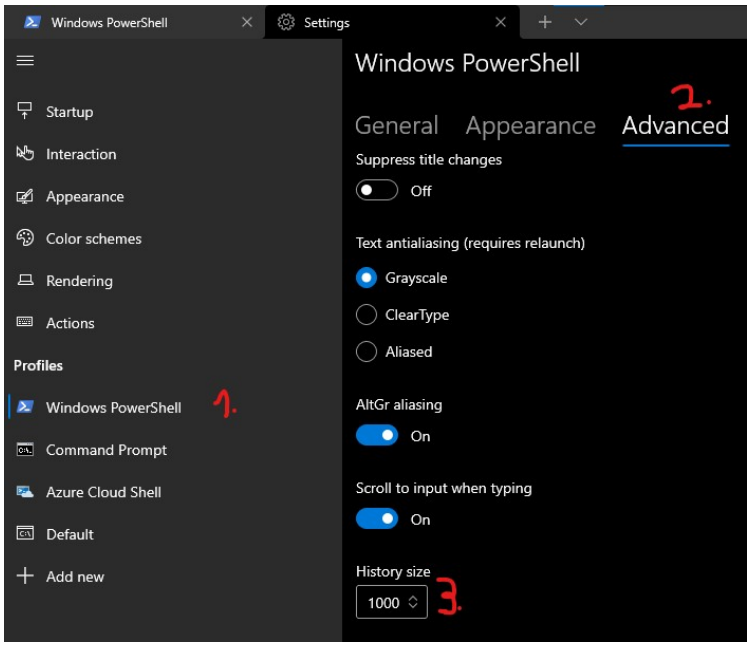
Obrázek IV: *Otevření nastavení*

Přesuneme se do nastavení naší konzole do kolonky „Appearance“, kde nastavíme font písma v konzoli na 8 (viz *Obrázek V*). Změny potvrdíme modrým tlačítkem v pravém dolním rohu.



Obrázek V: *Nastavení fontu písma*

Dále pokračujeme v nastavení, kdy se přesuneme do vedlejší kolonky „Advanced“. Zde změníme počet řádků, které si bude konzole pamatovat na 1000 (viz). Změny potvrdíme modrým tlačítkem v pravém dolním rohu.



Obrázek VI: *Úprava historie konzole*

Stiskněme klávesu F11 a v této fázi je čas hru spustit. Pomocí příkazů „cd *nazev_složky“* se dostaneme až k samotné hře a nyní zadáme příkaz „java –jar Prsi.jar“ a potvrdíme klávesou Enter. Hra se spustí automaticky.

8. Závěr

Projekt obsahuje vše, co bylo mnou očekáváno, vytvořil jsem hru, která někomu snad zpříjemní dlouhou chvíli. Jsem rád, že jsem poznal, jak se vytváří i rozsáhlejší projekt a hlavně, jak je třeba při jeho tvorbě přemýšlet.

Nejnáročnější části byly vytvoření designu karet a samotná logika hry a to hlavně díky jejich časové náročnosti.

Je jen jedna věc, kterou bych zkritizoval na svém projektu, čímž je mazání obrazovky. Nenalezl jsem lepší funkční řešení, než je nyní implementované.

V plánu mám aktuální řešení Prší převést do podoby JavaFX, přidat možnost hrát online pomocí serveru, zefektivnit míchání karet a dále bych rád zprovoznil také možnost dohrávat až do konce hry. Tedy tak, aby hra určila všechny stupně vítězů.

9. Bibliografie

ANSI escape kód. (2. květen 2021). Načteno z Wikipedia: https://cs.wikipedia.org/wiki/ANSI_escape_k%C3%B3d

Get Windows Terminal. (2. květen 2021). Načteno z Microsoft Store: https://www.microsoft.com/en-us/p/windows-terminal/9n0dx20hk701?activetab=pivot%3Aoverviewtab

Java Program to Find the Largest Number in an Array. (2. květen 2021). Načteno z Sanfoundry: https://www.sanfoundry.com/java-program-find-largest-number-array/

NetBeans. (2. květen 2021). Načteno z Wikipedia: https://cs.wikipedia.org/wiki/NetBeans