Zápočtový program – Dokumentace

Programování II (NPRG031)

LS 2022/23

Le Duc Hung

Zadání práce

Zadáním práce bylo vytvořit klasickou karetní hru s použitím jazyka C# s grafickým rozhraním (Windows Forms).

Hra je určena pro jednoho hráče, přičemž hráč hraje proti počítači, který má implementované rozhodovací procesy. Hráč a počítač se v tazích střídají, přičemž na začátku hry vždy začíná hráč. Po zahrání nějaké karty hráčem se počítač rozhodne, zda bude hrát nějakou platnou kartu (dle pravidel prší) nebo si nějakou bude tahat.

Pravidla hry

Hra začíná s 4-6 kartami, přičemž se hraje s klasickými Mariášovými kartami s hodnotami (7, 8, 9, 10, Spodek, Eso, Svršek, Král) a barvami (srdcové - ♥, žaludové - ♣, kulové - ♦, listové - ♠).

Program respektuje klasická pravidla Prší, tj. na položenou kartu lze hrát jen kartou, která má shodnou hodnotu nebo barvu, výjimka je karta Svršek, kterou lze zahrát na jakoukoliv kartu.

- Pokud jeden z hráčů zahraje sedmu, tak si druhý hráč, buď tahá 2 karty, nebo ji může přebít jinou sedmou (počet karet k tahání se vždy zvýší o 2).
- Pokud jeden z hráčů zahraje Eso, tak druhý hráč, buďto stojí, nebo položené Eso přebije svým Esem. Jestliže jeden z hráčů stojí, netahá si karty.
- Pokud jeden z hráčů zahraje Svršek, tak vybírá novou barvu, na kterou musí protihráč hrát.

Naimplementoval jsem také speciální pravidlo, tj. červená sedma vrací do hry – jestliže jeden z hráčů položí poslední kartu a je to karta červená nebo sedma, tak protihráč, pokud má v ruce červenou sedmu, může hráče vrátit zpět do hry (tahá si příslušný počet karet po položení srdcové sedmy).

Vítězem je hráč, kterému již nezbývají žádné karty.

Průběh hry

Na začátku se zobrazí toto okno a hráč si vybírá, s kolika kartami se začíná hrát:

Po vybrání počtu karet se rozdají karty a zobrazí se toto okno:

V horní části se nachází prostor pro zprávu, kterou vám počítač sdělí po odehrání tahu.

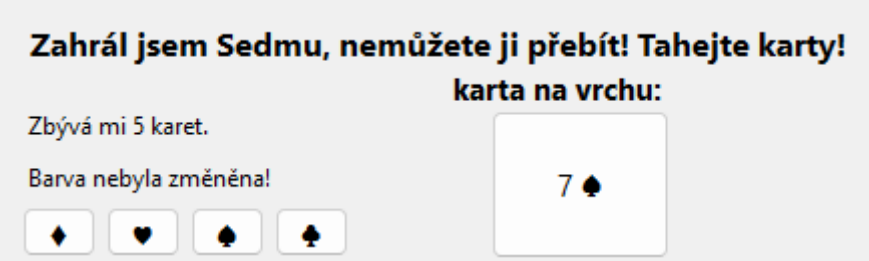
Dále tu máme počet zbývajících karet počítače, zprávu, zda někdo změnil barvu a předchozí kartu, která byla odehrána.

Tlačítka:

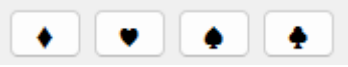
- 4 tlačítka s barvami, která slouží pro změnění barvy po položení Svršku (jen pro hráče)
- Tlačítko, které reprezentuje kartu na vrchu hracího balíčku (nemá žádnou speciální funkci po kliknutí)
- Tlačítko „tahat karty“ – tahání karet (ve speciálních případech se jeho text změní a slouží k účelům vhodným v daných případech)

- V dolní polovině okna se nachází karty hráče, tato část se zaplňuje zleva, a to shora dolu

Poté, co hráč zahraje nějakou kartu nebo nějakou tahá, hraje počítač a zobrazí informační zprávu, př.:



Speciální situace:

1. Pokud hráč zahraje Svršek, vybírá si barvu, na kterou chce změnit barvu hry, tj. vybírá si jednu z těchto barev:
   
2. Jestliže počítač zahrál Eso a hráč žádné Eso v ruce nemá, tak mu počítač sdělí, že stojí a musí kliknout tlačítko „pokračovat“, tj. změněné tlačítko pro tahání karet.
3. Jestliže hrát nemá žádnou sedmu k přebití položené sedmy, tak klikne tlačítko „tahat karty“ a přidá se mu do rukou příslušný počet karet.

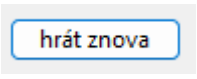
Na konci hry:

1. Pokud vyhraje hráč, vypíše se tato zpráva:
   Vyhrál jste!
   Konec hry!

2. Pokud vyhraje počítač, vypíše se tato zpráva:
   Byl/a jste poražen/a.
   Konec hry!

3. Pokud žádný z hráčů nemůže hrát a dojdou karty, vypíše se tato zpráva:
   Došly karty!
   Konec hry!

Na konci hry se vždy zobrazí toto tlačítko, které vás vrátí na úvodní okno:



Program

Jelikož byl program vytvořen pro Microsoft Forms, tak se nachází v souborech Form1.cs, kde je zdrojový kód, a dalších, jako je soubor Designer pro rozpoložení jednotlivých komponentů v okně.

Nejprve jsem se snažil vyřešit přípravu hry, tj. rozdání a tahání karet. Dále jsem vytvořil nějaké pomocné funkce, jako je kontrola validity pokládání karet, a nakonec jsem řešil samotnou logiku hry tak, tj. tah hráče a počítače, aby byla v souladu s pravidly hry.

Kód se nachází ve třídě Form1 odvozené od třídy Form.

Všechny funkce a proměnné jsou podrobněji popsány komentáři v souboru hry (Form1.cs).

Pomocné funkce:

a) `void NastavPocetKaretPocitace()`
- informuje hráče o počtu karet, který zbývá počítači
- implementováno i české skloňování

b) `void TahejKartu(List<(string, string)> tahajici)`
- pokud došly karty v tahacím balíčku, tak se obrátí balíček pro odkládání a nastaví se jako tahací balíček
- dále pokud nedošly karty, tak se tahá karta

c) `void VytvorKarticky(ref List<(string, string)> balicekKaret, List<string> barvy)`
- vytvoří se kartičky a vloží se do tahacího balíčku

d) `void ZamichejKarty(List<(string, string)> balicekKaret)`
- pomocí Fisher-Yates algoritmu se zamíchá balíček

e) `void VytvorHraciKarty(int vyskaKarty, int sirkaKarty, int pocetKaret, List<(string, string)> balicekKaret, List<Button> tlacitka, List<(string, string)> kartyHrace)`
- vytvoření tlačítek (32), která slouží jako kartičky hráče
- zobrazí se jen ty, které mají nějakou hodnotu, jinak se zneviditelní

f) `List<(string, string)> RozdejKarty(int pocetKaret)`
- Vrací seznam karet
- Pro rozdání karet počítači

g) `bool HratelnaKarta((string hodnoty, string barvy) co, (string hodnoty, string barvy) sCim)`
- parametry – 2 karty
- porovnává se první karta s druhou
- pokud někdo změnil barvu, tak se musí porovnávat s novou barvou a hodnotou druhé karty
- jinak se porovnávají dvě karty mezi sebou
- Svršek lze hrát kdykoliv

h) `void PolozitKartuNaVrch((string hodnoty, string barvy) karta, ref List<(string, string)> balicek, int indexOdebrani)`

- položí se karta na vrch
- ta se vloží do balíčku pro odkládání
- zobrazí se text karty na vrchu
- položená karta se odebere z balíčku toho, kdo ji zahrál

i) `void SedmaSrdcovaVraciDoHry(ref List<(string, string)> prohledavanyBalicek, ref List<(string hodnoty, string barvy)> kartyVracejicihoSe)`

- řeší případ, kdy hráč nebo počítač již nemá žádné karty, potom pokud ten, který je stále ve hře, má u sebe Sedmu srdcovou a lze ji hrát, tak se ten, kterému došly karty, vrací do hry a tahá si příslušný počet karet
- zobrazí se zpráva, která informuje hráče

j) `void HledaniVKartachPc(string cimZacit, string hledanaBarva, ref bool pokracovat, string specifikovane = "")`

- určena pro počítač
- slouží k vyhledání validní karty, kterou by mohl počítač zahrát
- cimZacit určuje, zda chceme hledat v hodnotách či barvách karet
- hledanaBarva – aktuální barva
- pokracovat – tato proměnná je využita v logice počítače, viz. Zdrojový kód
- podrobnější info. Viz. Program

k) `void NastavitNovouBarvu(string barva)`

- po tom, co hráč klikne na tlačítko pro změnu barvy, tak se provede tato funkce
- nastaví se nová barva
- zobrazí se zpráva
- provede se tah počítače

Jádro programu (switch):

`void NastavStav(Stav novy)`

a) `case Stav.START:`

- na začátku, zobrazí úvodní obrazovku

b) `case Stav.START:`

- připraví obrazovku hry
- připraví všechny potřebné náležitosti pro začátek hry

c) `case Stav.TAH_HRACE:`

- po tom, co hráč ukončí svůj tah, tak chceme aktualizovat karty, které má hráč v ruce
- karty hráče (tlačítka) se přepíší, aby byla uspořádaná a měla správnou hodnotu

d) `case Stav.TAH_PC:`

- implementován tah počítače
- kontrola, zda nedošly karty
- kontrola, zda nemá končit hra

e) `case Stav.KONEC:`
- zobrazí se obrazovka konce hry
- nastaví se výchozí hodnoty proměnných

Metody tlačítek:

a) `private void Karticka_Click(object? sender, EventArgs e)`
- chování, při kliknutí karty hráče
- lze-li kliknutá karta hrát, tak se zahraje a pokud se zahraje jedna ze speciálních karet, tak se hra chová podle pravidel
- na konci se provede tah počítače

b) `private void tahat_karty_Click(object sender, EventArgs e)`
- řeší tahání jedné nebo více karet
- pokud počítač zahrál Eso, tak se text tohoto tlačítka změní na „stát“ a po kliknutí hráč stojí
- text se mění i na „pokračovat“, podrobnější info. viz. Program

Rozhodovací procesy počítače

Rozhodovací procesy počítače jsem implementoval takto:

1) Počítač nejprve kontroluje, zda byla položena sedma nebo eso. Pokud má u sebe nějakou kartu, kterou může jednu z těchto karet přebít, zahraje takovou kartu, na kterou narazí pří prohledávání svých karet jako první. Pokud žádnou nemá, tahá si takový počet karet, který je nutný, nebo v případě položeného esa stojí.
2) Dále, pokud je položen svršek, tak se počítač nejprve snaží zahrát kartu s barvou shodnou se změněnou barvou, jinak se pokusí zahrát svršek, pokud nějaký u sebe má. Pokud u sebe nemá žádnou z těchto karet, tahá si kartu.
3) Zahraje-li počítač svršek, změní barvu hry na barvu první karty, kterou má ve svém balíčku
4) Pokud má hráč poslední kartu, počítač se snaží zahrát sedmu nebo eso
5) Jinak, pokud není položena žádná speciální karta, počítač nejprve hledá ve svém balíčku karty podle barvy, potom podle hodnoty karty na vrchu balíčku pro odkládání, případně pokud žádnou kartu nenalezne, hledá svršek (ten může zahrát na jakoukoliv kartu)
6) Pokud nemůže zahrát žádnou kartu, tahá kartu z hracího balíčku
7) Zahraje-li poslední kartu a je to svršek, tak vždy změní barvu hry na kule

Použité datové struktury

V programu jsou využity:

- enum pro stav hry a ukládání, kdo je na tahu
- seznam pro karty v „rukách“ hráče a počítače, je také použit jako fronta

Průběh práce

Při dlouhém rozmýšlení nad tématem jsem se rozhodl pro hru a to Prší. Jelikož jsme druhý semestr probírali téma GUI, tak jsem se rozhodl, že bych tuto hru mohl implementovat s grafickým rozhraním. Co se týče náročnosti implementace, tak si myslím, že nebyla příliš těžká, ale ani ne příliš lehká. Určitě tu byly chvíle, kdy šlo psaní programu hladce, ale taky tu byly chvíle, kdy jsem tápal nad problémy a bugy, které se objevovaly.

Dost značnou část času na začátku jsem věnoval rozmýšlení nad vzhledem oken a rozpoložení jednotlivých komponentů v nich. Dále jsem krok po kroku přidával další a další funkce. Jak jsem zmínil výše, tak se během psaní programu vyskytovaly problémy. Nevyskytovaly se příliš často, ale když už ano, tak to někdy trvalo až několik hodin, než jsem dal vše znova do pořádku.

Nakonec jsem vše úspěšně dokončil a doufám, že program funguje, jak má.