STŘEDOŠKOLSKÁ ODBORNÁ ČINNOST

Obor SOČ: 18. Informatika

Název práce: Tvorba karetní hry v jazyce C#

Vypracoval: Martin Kovář

Škola: Střední škola spojů a informatiky, Tábor, Bydlinského2474

Kraj: Jihočeský

Konzultant: Ing. Dana Almášiová

Prohlašuji, že jsem svou práci SOČ vypracoval samostatně a použil (a)jsem pouze podklady (literaturu, projekty, SW atd.) uvedené v seznamu vloženém v práci SOČ. Prohlašuji, že tištěná verze a elektronická verze soutěžní práce SOČ jsou shodné.

Nemám závažný důvod proti zpřístupňování této práce v souladu se zákonem č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) v platném znění.

V ……dne…… ……

Podpis

1 Obsah

2 Anotace

Cílem této práce bylo vytvoření karetní hry v programu Visual Studio 2019. Hra je vytvářena v programovacím jazyce C#. Práce popisuje tento programovací jazyk C#, vývojářské prostředí Visual Studio, řešení grafiky aplikace a řešení postupu hry. Výsledek práce slouží jako aplikace s hrou hráče proti krupiérovi-počítači.

Součástí práce je funkční aplikace v podobě spustitelné verze, která se nachází na přiloženém nosiči.

3 Klíčová slova

Visual Studio 2019, jazyk C#, generátor náhodných čísel, podmínky, třídy, datagrid, soubor

Úvod

V této práci se věnuji tvorbě karetní hry v programu Visual Studio 2019. Hra je vytvářena v programovacím jazyce C#.

Původní inspirací mi byla karetní hra Blackjack, kterou často hraji s přáteli. Cíl mé práce bylo vytvořit zábavnou hru, ve které hraje hlavní faktor štěstí a také včasné ukončení svého tahu. Nesmí chybět ani výsledkové porovnání všech výherců.

Avšak hlavním úkolem, který jsem si dal, bylo vytvořit hru bez chyb a možného obcházení hry za možností získat lepší skóre.

Také chci popsat programovací jazyk C#, zmínit jeho distribuce, vývojářský program Microsoft Visual Studio a jeho prostředí, řešení karetní hry z pohledu grafické i kódové stránky a definovat některé podmínky a třídy.

Toto téma jsem si vybral za účelem donucením se získávat nové informace a zkušenosti z oblasti programování z důvodu vytvoření zábavné a bez chybné hry.

4 O jazyce C#

Je to objektivně orientovaný programovací jazyk, který dokáže vytvářet mnoho typů bezpečných vysoce náročných aplikací. Zakládá se na programovacích jazycích C++ a Java. Využívá se k tvorbě formulářových aplikací, webových aplikací, mobilního softwaru a databázových programů. Zdrojový kód mívá příponu .cs.

Platforma .NET je virtuální spouštěcí systém nazvaný modul CLR (Common Language Runtime) a sada knihoven tříd. Základem prostředí pro spuštění a vývoj je rozhraní příkazového řádku, ve kterém spolupracují jazyky a knihovny.

C# (C Sharp) byl vyvinut firmou Microsoft v roce 2000. Spolu s ním byla vyvinuta platforma .NET Framework. První vydanou verzí bylo C# 1.0, s NET Frameworkem 1.0 obsahovala základní podporu objektového programování. Další verze 1.1 a 1.2 se objevily s Visual Studio .NET 2003, s Visual Studio 2005 přišla verze 2.0. Aktuální verze Visual Studio 2019 podporuje verzi 8.0, která je zpětně kompatibilní.

4.1 Distribuce pro jazyk C#

Pro jazyk C# existuje mnoho distribucí. Mezi nejznámější patří MonoDevelop, SharpDevelop, Xamarin Studio, Visual Studio for Mac, Baltík, vyvinutý českou společností SGP Systems a Microsoft Visual Studio, využívaný při této práci.

4.2 Microsoft Visual Studio

Vývojové prostředí od Microsoftu bývá využíváno v konzolových aplikací s grafickým rozhraním spolu s aplikacemi Windows Forms.

Společnost Microsoft poprvé vydalo Visual studio v roce 1997. Hlavním cílem bylo spojit více svých programovacích nástrojů dohromady a použít jedno vývojové prostředí pro více jazyků.

Významnou proměnu se Visual Studio dočkalo v únoru 2002, kdy bylo představeno vývojové prostředí pro spravovaný kód za použití .NET Framework. Název se tedy změnil na Visual Studio.NET.

Na NET framework se lze dívat jako hierarchickou kostru Visual Studia. Zajišťuje automatickou podporu překladu do více programovacích jazyků, například C++ nebo Jscript. Vestavěná funkce CLS zajišťuje překlad do mezijazyka.

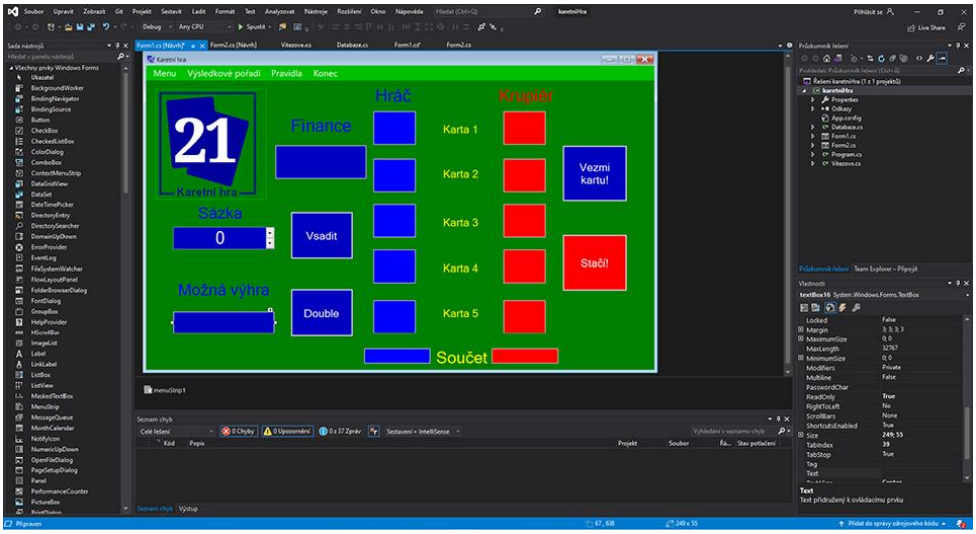
V říjnu 2005 došlo k odstranění přípony .NET, i přesto se stále zaměřuje na .NET Framework. Stalo se tak ve verzi Visual Studio 2005.

Aktuální verzí na trhu je Visual Studio 2019. Vydáno Microsoftem bylo 2. dubna 2019.

5 Visual Studio 2019

Při této práci jsem využíval Visual Studio Community 2019. Po spuštění aplikace a před vytvořením nového projektu vybíráme šablonu. Pro svojí aplikaci jsem zvolil *Aplikace Windows Forms (.NET Framework).* Následuje název projektu, umístění, název řešení a architektura, kterou jsem zvolil *.NET Framework 4.7.2.* Po vytvoření aplikace se spustí okno designera (*Form1*), která představuje vizuální stránku aplikace.

5.1 Prostředí



V tomto formuláři se dostávám k designovému oknu hry. Je zde také sada nástrojů, která se nachází v levém okraji obrazovky a byla již využita v oknu designera (*Form1.cs[Návrh]*).

Pravá dolní část (*Vlastnosti*) zobrazuje možnost *Vlastnosti* (ikona papíru a mechanického klíče), nebo *Události* (ikona blesku), pro jakýkoliv určený nástroj, nebo celé designové okno.

V pravé horní časti se nachází *Průzkumník řešení*, ve kterém můžeme spouštět jednotlivé součásti projektu.

Možnost aplikaci spustit najdeme pod tlačítkem *Spustit*, nebo klávesou F5.

6 Form1

Form1.cs[Návrh] je nastaven ve *StartPosition* na *CenterScreen*, aby aplikace byla vždy uprostřed obrazovky.

Panel je pojmenován *karetniHra*. Jeho pozadí tvoří zelená barva a má vlastní ikonu tvořenou v rastrovaném programu GIMP.

Ve *Form1.cs[Návrh]* se nachází 11 labelů, 14 textBoxů, 4 buttony, numericUpDown a pictureBox. Nechybí zde ani menuStrip, který tvoří hlavní menu celé hry.

6.1 Členské proměnné

Jako první věc při tvorbě této hry bylo potřeba nadefinovat si některé členské proměnné. Proměnná slouží k definici a ukládání dat. Během běžícího programu je lze také měnit pracovat s nimi a následně ukládat. Členská proměnná se může využívat v celém formuláři, jelikož se deklaruje ještě před třídou *Form1()*. Aby s nimi program mohl pracovat, musí se nejprve deklarovat (vytvořit).

```
 //Generátor náhodných čísel
Random náhoda = new Random();
//Počet karet Hráče
int pocetkarethrac = 0;
//Počet karet Krupiéra
int pocetkaretkrupier = 0;
//Součet Hráče
int soucethrac = 0;
//Součet Krupiéra
int soucetkrupier = 0;
//Finance hráče
int finance = 500;
//možná výhra
int moznavyhra = 2;
//karta hráče
int kartahrac = 0;
//sázka
int sazka = 0;
//Databáze
private Databaze databaze;
//Počet her
int pocether = 0;
```

Ke generování náhodných čísel využiji třídu *Random* a dále ji budu definovat jako *náhoda*. Třída *Random* generuje pseudonáhodné číslo, které vypočítá na základě algoritmu.

Dále se zde nacházejí další číselné členské proměnné, které během hry budu potřebovat a využívat. Jsou definovány pouze celé číselné hodnoty, tedy *int*.

```
InitializeComponent();
databaze = new Databaze("vitezove.csv");
```

Privátní databáze slouží k odkazu na vytvořenou třídu *Databaze.cs* a nyní jako novou instanci *databaze*. Třída *Databaze.cs* slouží k ukládání dat do formátu CSV. Bude se využívat pro ukládání vítězů, jejich financí a počtu odehraných her.

6.2 Vsazení

Před rozehráním každé hry, si hráč musí vsadit určitou částku, kterou je nutné zapsat do *numericUpDown1*. Tento ovládací prvek slouží k získávání nebo nastavení hodnoty pro zvýšení nebo snížení číselníku.

```
//Vsazení
sazka=Convert.ToInt32(numericUpDown1.Value);
finance = finance - sazka;

if (finance < 0)
    {
       MessageBox.Show("Nelze jít do mínusu");
       finance = finance + sazka;

     }
else
     {
     //zapsat možnou výhru
     moznavyhra = sazka * 2;
     textBox16.Text= moznavyhra.ToString();

     //Připsat základ hráči
     textBox13.Text = finance.ToString();
```

Po stisknutí tlačítka *Vsadit (button3)* je sázka odečtena od financí a neprodleně poté se možná výhra zapíše do *textBox16*. Ta je tvořena dvojnásobkem sázky. Ale nejprve dojde ke kontrole, pomocí jednoduché podmínky *if*, jestli po odebrání sázky od financí nedojde k záporné hodnotě a jelikož je to v této hře zakázané, dojde pomocí *messageBoxu* k vypsání, že nelze jít do záporných hodnot.

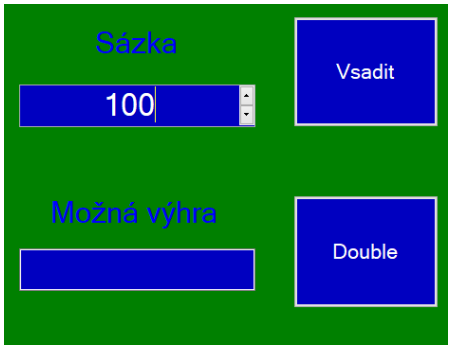
Podmínka *if*, za kterou následuje logický výraz, je buď pravdivá, provede se zadaný příkaz, nebo nepravdivá, následující příkaz se přeskočí a pokračuje se až pod ním. V případě, že se podmínka neprovede, může následovat slovo *else*, které vykoná své zadané příkazy.

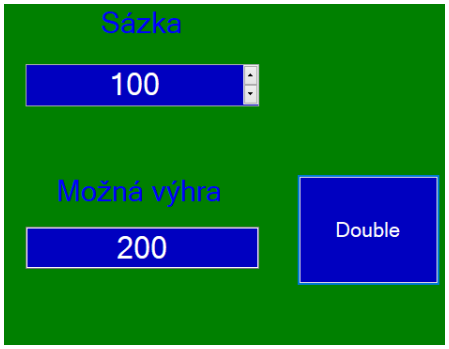
Když je sázka ukončena a schválena, nastaví se *numericUpDown1* na *ReadOnly,* tak aby do něj již nešlo zapisovat a zmizí *button3,* aby nyní nebylo možné vsadit. Oproti tomu se objeví *button1* a *button2*, pomocí kterých můžeme sesílat karty (čísla) na hrací plochu.

```
//už nemůžeš vsázet
numericUpDown1.ReadOnly = true;

//zmizení buttonu 3
button3.Visible = false;

button1.Visible = true;
button2.Visible = true;
```





Při stisknutí tlačítka také dochází k navýšení hodnoty u počtu her a to vždy o jednu navíc.

```
pocether++;
```

6.3 Vezmi kartu! (button1)

```
kartahrac = náhoda.Next(1, 12);

switch (pocetkarethrac)
{
case 0:
          // Vezmi první kartu
         textBox1.Text=kartahrac.ToString();
          soucethrac = kartahrac;
          pocetkarethrac = 1;
       break;
```

Nyní využiji generování náhodných čísel od 1 do 11, aby byla karta (číslo) vždy náhodná.

Příkaz *switch* slouží k větvení programu do více větví a v mé hře ho využiji vždy po stisknutí tlačítka *Button1* k postupnému výkladu jednotlivých karet (čísel) na hrací plán. Čísla budou zapisována do jednotlivých textBoxů. Každá větev za klíčovým slovem *case*, která obsahuje nějaký příkaz nebo blok příkazů, musí být zakončená příkazem *break;*.

Po každém stisknutí tlačítka *Button1* bude program sčítat celkový součet hodnot karet (čísel) a bude měnit počet vzatých karet, podle toho kolikátou kartu si hráč vezme.

```
soucethrac = soucethrac + kartahrac;
pocetkarethrac = 2;
```

Těchto karet bude možno vzít pouze pět, poté dojde pomocí *MessageBoxu* k upozornění, že další kartu nelze vzít.

```
case 5:
     // Už nemůžeš vzít kartu
     MessageBox.Show("Už nemůžeš vzít kartu");
     break;
```

Po každé vzaté kartě se celkový součet karet (čísel) bude vypisovat do *textBox6*, tak aby hráč vždy věděl jakou hodnotu má a popřípadě rozmýšlel svůj další tah.

```
//vypsání součtu čísel
textBox6.Text = soucethrac.ToString();
```

Pro kontrolu celkového součtu, tedy aby nebyla překročena hodnota 21 využiji opět podmínku *if*.

```
 //zjištění jestli nebylo překročena hodnota 21
if (soucethrac > 21)
{
  //Napsat že hráč prohrál
  MessageBox.Show("Prohrál jsi");

  //Připrav novou hru v případě, že hráč překročil 21
  novaHra();

  //Už nemmáš žadné finance, ukonči hru
  if (finance == 0)
  {
      MessageBox.Show("Už nemáš žádné finance, prohrál jsi");
      button1.Visible = false;
      button2.Visible = false;
      button3.Visible = false;
      button4.Visible = false;
```

```
    }
}
```

Jestliže hráč překročil vyšší hodnotu než 21 *MessageBox* vypíše, že prohrál a vsazenou částku ztrácí. K tomu slouží a k obnovení hry, kromě financí, počtu odehraných her a jména, vyvolaná metoda *novaHra()*. Tato metoda byla vytvořena tak aby po ukončení hry, bylo vše, kromě financí, počtu odehraných her a jména, přepsáno do původní podoby.

Kontrolu celkového stavu konta hlídá druhý *if* v pořadí. Pokud se počet financí bude rovnat nule, *MessageBox* vypíše, že hráč nemá žádné finance, nemá již co vsadit, a tedy prohrává celou hru. Poté zmizí všechny buttony, tak aby hráč byl nucen, pomocí menu, opustit hru nebo začít hru novou.

6.4 Double

Kód tlačítka *button4* je tvořen převážně podmínkami *if*. Tlačítko je možné využít až po seslání druhé karty(čísla) na hrací plochu.

```
if (pocetkarethrac == 0 || pocetkarethrac == 1 )
{
    MessageBox.Show("Musíš mít aspoň dvě karty");
}
```

Pokud je tlačítko Double, využito předčasně, *MessageBox* vypíše, že hráč musí mít alespoň dvě karty. Tuto podmínku opatřuje *if*. V případě, že podmínka byla splněna, možná výhra se zdvojnásobí, ale hráči již není dovoleno brát další karty a musí kliknout na tlačítko Stačí! (*button2*). Další podmínka *if* zajišťuje aby finance po stisknutí tlačítka nebyly záporné.

```
if (pocetkarethrac == 4)
{

  kartahrac = náhoda.Next(1, 12);
  textBox5.Text = kartahrac.ToString();
  soucethrac = soucethrac + kartahrac;
  pocetkarethrac = pocetkarethrac + 1;

  //vypsání součtu čísel
  textBox6.Text = soucethrac.ToString();
}
```

Stejnou podmínkou je řešeno i kolikátá karta (číslo) byla seslána na hrací plochu, kvůli počtu karet. Opět využiji generování náhodných čísel od 1 do 11, aby byla karta (číslo) vždy náhodná. Stejně jako u tlačítka *button1 (Vezmi kartu)* je zde řešeno nepřekročení hodnoty 21 a ověření

nezáporných hodnot ve financí. V případě, že podmínka projde zmizí *button4* a *button1*, aby na hrací ploše zbyl pouze *button2 (Stačí!)* a hráč musel tedy ukončit svůj tah.

6.5 Stačí! (button2)

Poslední button je kódem rozsahově nejdelší. Základ opět tvoří podmínky *if*. Karty (čísla) jsou postupně sesílány na hrací plochu, přičemž podmínka if hlídá jestli byla překročena hodnota 16 a zároveň nebyla překročena hodnota 21.

Opět využiji generování náhodných čísel od 1 do 11, aby byla karta (číslo) vždy náhodná.

Když krupiér přesáhne hodnoty 16, vypíše se součet hodnot karet do *textBox12,* končí svůj tah a porovnávají se součty hodnot karet z *textBox6* a *textBox12*. V případě, že krupiér přesáhne hodnoty 21, je možná výhra přičtena k financím a hodnota v *textBox13* se přepíše. Po připsaní výhry, *messageBox* vyhlásí vítěze. To platí i v případě, že součet hodnot z *textBox6* bude vyšší než z *textBox12*.

```
  if (soucethrac == soucetkrupier)
        {
  //pokuď má Hráč míň karet napiš:"Vyhrál hráč, protože má míň
karet"
  if (pocetkarethrac < pocetkaretkrupier)
  {
    //Připsat výhru Hráči
    finance = finance + vyhra;

  //Napsat že Hráč vyhrál
  MessageBox.Show("Vyhrál jsi " +vyhra+ ", protože máš míň
karet");
```

Jestliže budou v *textBox6 a textBox12* stejné hodnoty, podmínka *if* porovná počty zahraných karet a ten kdo bude mít méně karet vyhrává. V případě hráčovy výhry je možná výhra přičtena k financím a hodnota v *textbox13* je tedy přepsána. MessageBox vypíše vítěze a vyhranou částku.

Může nastat situace, kde hodnoty v *textBox6 a textBox12* budou stejné a počet karet u krupiéra i u hráče bude také stejný, proto jsem si připravil předposlední *if*.

```
if (pocetkarethrac == pocetkaretkrupier)
    {
    //Připsat sázku hráči
    finance = finance + (vyhra / 2);

    MessageBox.Show("Remíza, vrací se ti sázka");
    }
```

Tato podmínka vyřeší vrácení sázky do *textBox13* a *MessageBox* vypíše zadaný text.

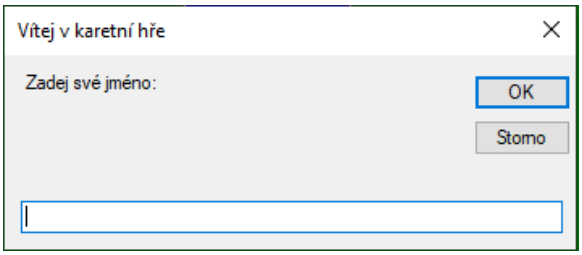
Po dohrání hry se připraví nová hra pomocí vyvolané metody *novaHra()* a podmínka *if* zkontroluje jestli hráč nedosáhl nulové hodnoty ve financích.

6.6 Načtení formuláře

Při každém načtení formuláře a před jeho zobrazením je potřeba připravit novou hru.

```
  private void karetniHra_Load(object sender, EventArgs e)
  {
    string jmeno = Interaction.InputBox("Zadej své jméno:",
"Vítej v karetní hře", "");
    if (jmeno != "")
     {
      label1.Text = Convert.ToString(jmeno);
      }
       //Připrav novou hru
       finance = 500;
       numericUpDown1.Value = 10;
       textBox13.Text = finance.ToString();
  }
```

Jako první je připraven *InputBox,* ale nejprve si do *using* přidám *Microsoft VisualBasic*. *InputBox* zobrazuje výzvu v dialogovém okně a čeká, než uživatel zadá text a po stisknutí tlačítka vrací řetězec obsahující obsah textového pole. V mé hře slouží k zadávání jména. Obsah textového pole je konvertován do *label1* a jeho veškerý text je nahrazen.



Finance jsou nastaveny na základní částku 500 a předepsána je i možná sázka na 10.

7 Menu hry

Menu celé hry tvoří *menuStrip1*. Obsahuje uložení hry, načtení hry, novou hru, výsledkové pořadí, pravidla a dvě tlačítka pro ukončení hry.

7.1 Ukládání

Ukládá se dvojím způsobem, ale nejprve si do *using* přidám *Systém. IO*.

```
using System.IO;
```

První způsob ukládání je do textového formátu s příponou *txt!*. Text je v nich uložen jednoduše na jednotlivých řádcích. Ukládat do nich budu jméno hráče, finance a počet odehraných her, zvlášť, tak aby je při načtení hry šlo snáze načíst. Ale nejprve je vytvořena podmínka *if*, pro možné uložení hry, až nad částkou 999.

```
  //Uložení částky
  //otevření
  StreamWriter souborCastka = new
StreamWriter("ulozenaCastka.txt!");
  //Ulož
  souborCastka.WriteLine(textBox13.Text);
  //zavření souboru
  souborCastka.Close();
```

K zapisování do textových souborů nám .NET poskytuje třídu *StreamWriter* a v mém kódu je již nasměrován na správný soubor, který je uložen ve složce *Debug*. Částku si vezmu s *textbox13*, kde je uložena finanční hodnota. Soubor se uloží a zavře. Podobně to funguje i u jména hráče a počtu odehraných her.

Druhé ukládání bude sloužit k výsledkovému pořadí, ale nejprve jsem si vytvořil třídy *Vitezove.cs* a *Databaze.cs*. Ve třídě *Vitezove.cs* definuji jméno, počet odehraných her a finance. Konstruktor bude instanci inicializovat na základě těchto tří vlastností. Třída *Databaze.cs* obsahuje metody k ukládání obsahu databáze do souboru CSV. K zápisu opět využiji třídu *StreamWriter*, ale tentokrát v *using* bloku. Uvnitř pro každého vítěze vytvořím pole stringů z jeho jménem, počtem odehraných her a financemi.

// otevření souboru pro zápis

```
using (StreamWriter sw = new StreamWriter(soubor, true))
 {
  // projetí uživatelů
  foreach (Vitezove v in vitezove)
```

```
    {
    // pole hodnot
    string[]   hodnoty   =   {   v.Jmeno,   v.pHer.ToString(),
v.Body.ToString() };
    // vytvoření řádku
    string radek = String.Join(";", hodnoty);
    // zápis řádku
    sw.WriteLine(radek);
    }
```

Pro obnovení hodnot, tak aby vítěz nebyl pořád stejný, jsem vytvořil metodu *Obnov()*. Využívá se pro smazání obsahu databáze po uložení hry.

Jako poslední si ve *Form1* definuji jméno, počet odehraných her a finance. Program poté, po vyvolání metod, vítěze přidá do databáze, uloží a obnoví.

7.2 Načtení uložené hry

Načíst lze jen poslední uloženou hru pomocí třídy *StreamReader,* která vrací řádek textu ze souboru. Konkrétně přepíše *label1* na jméno uložené v souboru a změní se také počet her a finance.

```
  //otevření
  StreamReader           ulozenyHrac           =           new
StreamReader("ulozenyHrac.txt!");

  //doplnění Částky
  label1.Text = ulozenyHrac.ReadLine();

  //zavření souboru
  ulozenyHrac.Close();
```

Poté program připraví novou hru vyvolanou metodou *novaHra(),* tak aby z předešlé hry nebránilo načtenému hráči k postupu ve hře.

7.3 Nová hra

Jestliže hráč bude z jakéhokoli důvodu chtít začít hru od začátku, nemusí celou aplikaci vypínat a zapínat, ale poslouží k tomu v *menuStrip1 Nová hra*. Po stisknutí se nejprve objeví *InputBox,* do kterého je možné zadat jméno a poté vyvolaná metoda *novaHra()* obnoví hrací pole. Následně bylo nutné přepsat hodnoty u financí na 500 a počet her na 0.

7.4 Pravidla

V případě, že hráč nezná pravidla hry, může využít, stisknutím na *Pravidla* v *menuStrip1,* jejich výklad. K výpisu pravidel slouží *MessageBox***.**

7.5 Ukončení

Pro konec hry lze využít dvě možnosti. Po stisknutí první možnosti *Ukončit* se vyvolá metoda *Close(),* která pouze ukončí aplikaci, ale neuloží ji.

```
//ukončit program
  Close();
```

K tomu slouží možnost *Uložit a ukončit.* Funguje na stejném principu jako *Uložit hru*, akorát se na konci vyvolá metoda *Close().*

7.6 Výsledkové pořadí

Poslední tlačítko v *menuStrip1* poslouží k vyvolání druhého formuláře, který se bude zobrazovat jako dialog.

```
Form2 a = new Form2();
a.ShowDialog();
```

8 Form2

Form2.cs[Návrh] je nastaven ve *StartPosition* na *CenterScreen*, aby okno *Form2* bylo vždy uprostřed obrazovky.

Panel je pojmenován *Form2*. Jeho pozadí tvoří zelená barva a má vlastní ikonu tvořenou v rastrovaném programu GIMP.

Tento panel slouží k zobrazení a porovnání výsledků všech hráčů, kteří ve hře přesáhli hodnoty financí 999 a následně hru uložili.

Je tvořen z velké části *dataGridView1*, který slouží jako tabulka a nachází se zde i *button1*, po jehož stisknutí vás program vrátí zpět do hry.

Při načtení formuláře se načítají uložená data ze souboru *vitezove.csv* do *dataGridView1* pomocí třídy *StreamReader*. Poté se přečte hlavička a odstraní se hodnoty z tabulky vyvolanou metodou *Clear()*.

```
// Přečti hlavičku
string hlavička = soubor.ReadLine();

// Vymaž stávající hodnoty z tabulky
dataGridView1.Rows.Clear();
```

Následuje postupné načtení řádků a přidávání obsahu souboru do tabulky. Po dokončení se soubor zavře.

```
// Postupně načítej řádky a přidávej do tabulky
string řádek;
while ((řádek = soubor.ReadLine()) != null)
```

```
  {
   string[] hodnoty = řádek.Split(';');
   dataGridView1.Rows.Add(hodnoty);
  }
   // Zavři soubor
   soubor.Close();
```

Barevná a textová úprava hlavičky *dataGridView1* je také definována v kódu.

Zbývá už jen přidat vzestupně číselné hodnoty do sloupce *Pořadí*, seřadit podle financí nejlepší hráče a také upravit vzhled celé tabulky.

```
  //Přidej pořadí
  this.dataGridView1.Rows[e.RowIndex].Cells["sloupecPoradi"].Va
lue = (e.RowIndex + 1).ToString();

  //seřaď od nejlepšího výsledku
  dataGridView1.Sort(dataGridView1.Columns[2],
ListSortDirection.Descending);
```

Po stisknutí tlačítka *button1* se vyvolá metoda *Close()*, čímž se zavře okno *Form2* a je možné dále pokračovat ve hře.

9 Pravidla hry

Hra má prostá pravidla, které nejde obejít a je nutné se jimi řídit. Hráč (vy), který zadá jméno, jež ho zastupuje hraje proti krupiérovi. Hráč má vždy na začátku nové hry částku v hodnotě 500 korun českých. Poté musí vsadit libovolnou částku a následně se vypíše možná výherní částka.

Po vsazení může hráč zahájit svoji hru. Bere si postupně karty (znázorněné čísly) pomocí tlačítka *Vezmi kartu!*. Karty mají hodnotu 1-11 a jejich hodnota při seslání na hrací plochu je náhodná. Hráč si může vzít maximálně pět karet, ale nesmí přesáhnout hodnotu 21. Pokud hráč nechce využít plný počet karet, může u třetí a čtvrté karty svůj tah ukončit pomocí tlačítka *Stačí!*. Tlačítko *Double* slouží ke zdvojnásobení sázky a tedy i možné výhry, ale smí se využít, až po zahrání dvou karet. Pokud je toto tlačítko využito zdvojnásobí se sázka, ale následně je možno zahrát pouze jednu kartu. U všech těchto variant program stále sleduje jestli nebyla přesažena hodnota 21.

Následuje krupiérův tah. Automaticky si bere karty, ale musí přesáhnout hodnotu 16.

Vyhrává vyšší součet hodnot, pod podmínkou, nepřekročení hodnoty 21. V případě stejných hodnot v obou součtech, nastává sčítání karet na obou stranách. Výherce poté určí menší součet karet. Je-li i stejný součet karet nastává remíza. Při výhře hráče je připsána výhra na jeho účet. V případě, že vyhrál krupiér, vsazená částka je hráči odepsána z účtu. U remízy se sázka vrací na účet Hráče.

Tento účet nesmí být záporný, pokud Hráč dosáhne částky nula, už nemá co vsázet a hra pro něj končí. Postupem hraní si hráč může hru uložit, či načíst.

Po uložení hry se finance, počet her a jméno hráče ukládá do tabulky výsledkového pořadí, kde hráči mohou porovnávat své výhry.

Závěr

Cíl mé práce bylo vytvořit jednoduchou, avšak zábavnou karetní hru v jazyce C#.

Zprvu jsem si dal vysoký cíl a s nepříliš velkou zkušeností v programovacím jazyce C# jsem se trápil. Po pár neúspěšných pokusech jsem si uvědomil, že tato hra jde vytvořit logicky ze základů, které jsem se naučil ve škole.

Základy jsem tedy vytvořil, a poté jsem do hry začal přidávat prvky a funkce, které jsem se naučil ve svém volném čase. Díky tomu jsem se lépe naučil pracovat s funkcemi, třídami a s prvkem dataGridView.

Hra je určena pro jednoho hráče, který hraje proti počítači. V budoucnu bych rád zlepšil hru po grafické stránce.

Ve hře mohou hráči změřit své štěstí, zabavit se a poměřovat své výhry. Můj hlavní cíl vytvořit zábavnou hru, bez chyb a možného obcházení hry za účele získat lepší skóre, jsem snad splnil.

Zdroje

C Sharp. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2021-03-30]. Dostupné z: https://cs.wikipedia.org/wiki/C_Sharp

Prohlídka jazyka C#. *Prohlídka jazyka C#* [online]. microsoft: microsoft, 2021, 28. 01. 2021 [cit. 2021-03-30]. Dostupné z: https://docs.microsoft.com/cs-cz/dotnet/csharp/tour-of-csharp/

Baltík. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2021-03-30]. Dostupné z: https://cs.wikipedia.org/wiki/Balt%C3%ADk

Uložení objektů do CSV v C#. *Uložení objektů do CSV v C#* [online]. itnetwork [cit. 2021-03-30]. Dostupné z: https://www.itnetwork.cz/csharp/soubory/c-sharp-tutorial-ulozeni-objektu-do-textoveho-souboru-csv

Microsoft Visual Studio. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2021-03-30]. Dostupné z: https://cs.wikipedia.org/wiki/Microsoft_Visual_Studio

Příkazy break, continue, goto, if - else, switch. *Programujte* [online]. 2009 [cit. 2021-03-30]. Dostupné z: http://programujte.com/clanek/2009082900-c-9-lekce-prikazy-break-continue-goto-if-else-switch/

Podmínky (větvení) v C# .NET. *ITnetwork* [online]. [cit. 2021-03-30]. Dostupné z: https://www.itnetwork.cz/csharp/zaklady/c-sharp-tutorial-podminky-vetveni-if-switch

Metody (Průvodce programováním v C#). *Microsoft* [online]. 2021 [cit. 2021-03-30]. Dostupné z: https://docs.microsoft.com/cs-cz/dotnet/csharp/programming-guide/classes-and-structs/methods

Přílohy

Dokument práce ve formátu PDF

Projekt Microsoft Visual Studia Tvorba karetní hry v jazyce C#