Protokol k semestrální práci

**Předmět:** Úvod do programování a algoritmů

**Zadání:** S využitím Taylorova rozvoje vytvořte program pro výpočet přirozeného logaritmu.

**Vypracoval:** Jiří Ingr

**Kruh:** 159

# Úvod

Zadáním úkolu je naprogramování Konzolové aplikace v programovacím jazyku C#, který vypočítá přibližnou hodnotu přirozeného logaritmu. Program umožnuje uživateli zadat jak hodnotu, ve které chce přirozený logaritmus vypočítat, tak počet iterací, neboli počet členů Taylorova rozvoje, od čehož se odvíjí výsledná přesnost výpočtu. Program byl napsán programu Microsoft Visual Studio 2019.

# Řešení

Cílem bylo vytvořit přehledný program pro výpočet přibližné hodnoty logaritmu v zadaném bodě, který má ošetřené uživatelské vstupy a možnost opakování funkce programu po ukončení výpočtu.

Pro začátek jsem vypočítal několik členů Taylorova rozvoje přirozeného logaritmu a přišel na jednoduchost jeho výpočtu. Poté jsem začal převádět algoritmus do programovacího jazyka.

Nejprve jsem deklaroval vstupní proměnné a vytvořil možnost zadat uživateli proměnnou x (která určí bod, ve kterém se vypočítá hodnota ) a počet členů (znamenající pořadí nejvyššího vypočítaného členu rozvoje).

Ve výpočetní části jsem nejprve zajistil jednodušší a přesnější způsob výpočtu. V první kroku se omezí zadaná hodnota čísla tak, aby byla vetší než 1 (pokud je číslo menší, převrátí se), což zapříčiní počítaní pouze s kladnými čísly. Pro tuto část jsem vytvořil funkci *Prevraceni*, která zadané číslo převrátí nebo nechá v původním stavu. V dalším kroku jsem zadání podělil číslem tolikrát, než byla jeho hodnota menší než , a v každém kroku přičetl jedničku do proměnné *pocetE.* V této části jsem použil for-cyklus.

Ze vzniklé hodnoty jsem již vypočítal Taylorův rozvoj pomocí funkce *Tayloruv\_rozvoj*, v níž je implementována funkce *Tayloruv\_clen*, které podle zadaných údajů vypočítají přibližnou hodnotu . K výsledku se nakonec přičte proměnná *pocetE*. *P*okud bylo číslo převráceno, výsledku se změní znaménko. Následně se vypočítaná hodnota vypíše.

Nakonec jsem ošetřil uživatelské vstupy, jak ošetřením definičního oboru funkce, tak využitím podmínek *FormatException* (pro zadání „nezpracovatelné hodnoty“) a *Exception* (obecná podmínka) a vytvořil cyklus při dokončení výpočtu, který umožňuje vrátit se na začátek programu.

POZN.: Funkce *Tayloruv\_clen*, *Tayloruv\_rozvoj* a *Prevraceni* jsou součástí třídy *Funkce.*

# Kód programu

**Kód třídy program:**

using System;

namespace Semestralni\_konzolova\_aplikace

{

class Program

{

static void Main(string[] args)

{

// počáteční popis prgramu

Console.ForegroundColor = ConsoleColor.DarkBlue;

Console.WriteLine("Program na výpočet přibližné hodnoty funkce ln(x) v daném bodě (pomocí taylorova rozvoje).\n");

Console.ResetColor();

do // cyklus pro opakování funkce programu, pokud uživatel zvolí možnost dalšího výpočtu

{

// uživatelský vstup hodnoty x

string vstupX = "";

double x = 0;

bool testVstupuX = true;

while (testVstupuX)

{

// ošetření a přetypování vstupu hodnoty x

try

{

Console.Write("Zadejte hodnotu x (x > 0): ");

vstupX = Console.ReadLine();

x = double.Parse(vstupX);

// ošetření definičního oboru ln(x)

if (x > 0)

{ testVstupuX = false; }

else

{ // pokud je vstup mimo definiční obor ln(x) zobrazí se chybová hláška

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Pro tuto hodnotu není funkce ln(x) definovaná. D(f) = (0; nekonečno)\nZadejte hodnotu z definičního oboru.");

Console.ResetColor();

}

}

catch (FormatException) // ošetření formátu vstupu

{

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Hodna není ve správném formátu.\nZkuste to znovu.");

Console.ResetColor();

}

catch (Exception) // obecná podmínka

{

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Něco se nepovedlo.\nZkuste to znovu.");

Console.ResetColor();

}

}

Console.WriteLine(); // estetická mezera

// uživatelský vstup počtu členů rozvoje

string vstupPocetClenu = "";

int pocetClenu = 0;

bool testVstupuPocetClenu = true;

while (testVstupuPocetClenu)

{

// ošetření a přetypování vstupu počtu členů

try

{

Console.Write("Zadejte počet členů taylorova rozvoje: ");

vstupPocetClenu = Console.ReadLine();

pocetClenu = int.Parse(vstupPocetClenu);

// ošetření hodnoty počtu členů rozvoje

if (pocetClenu > 0)

testVstupuPocetClenu = false;

else

{ // chybová hláška se zobrazí, pokud uživatel zadá číslo <= 0

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Neplatné číslo.\nZadejte celé číslo vetší než 0.");

Console.ResetColor();

}

}

catch (FormatException) // ošetření formátu vstupu

{

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Hodna není ve správném formátu.\nZadejte celé číslo větší než 0.");

Console.ResetColor();

}

catch (Exception) // obecná podmínka

{

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Něco se nepovedlo.\nZkuste to znovu.");

Console.ResetColor();

}

}

// potvrzení uživatelského vstupu

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("\nVypočítám pomocí {0} členů taylorova rozvoje hodnotu ln({1}).", pocetClenu, x);

Console.ResetColor();

// pokračování k výpočtu (aby si uživatel mohl zkontrolovat své zadání)

Console.WriteLine("Pro pokračování stikněte Enter...\n");

while (Console.ReadKey().Key != ConsoleKey.Enter) { }

// VÝPOČET

// pokud x náleží (0; 1) převrátí se jeho hodnota

double mezivypocet1 = Funkce.Prevraceni(x);

bool prevraceno = true;

if (mezivypocet1 == x)

prevraceno = false;

// proměnná x se dělí eulerovým číslem, až než je menší než jeho hodnota

int pocetE = 0;

double e = Math.Exp(1);

double odmocninaE = 1.6488; // odmocnina z e

while (mezivypocet1 > odmocninaE)

{

mezivypocet1 /= e;

pocetE++;

}

// výpočet přibližné hodnoty pomocí taylorova rozvoje

double mezivypocet2 = Funkce.Tayloruv\_rozvoj(mezivypocet1, pocetClenu);

// k mezivýsledku se přičte proměnná pocetE

double mezivypocet3 = mezivypocet2 + pocetE;

// výsledek se převrátí zpět v případě, že se výše jeho hodnota převrátila

double vysledek;

if (prevraceno)

vysledek = -mezivypocet3;

else

vysledek = mezivypocet3;

// vypsání hodnoty uživateli

Console.Write("Hodnota ln({0}) s danou přesnotí je přibližně: ", vstupX);

Console.ForegroundColor = ConsoleColor.Green;

Console.WriteLine(vysledek);

Console.ResetColor();

// příkaz pro ukončení programu klávesou Esc

Console.WriteLine("\nPro ukončení porgramu stiskni klávesu Esc, Nový výpočet spustíte stiknutím libovolné jiné klávesy...\n");

} while (Console.ReadKey().Key != ConsoleKey.Escape);

}

}

}

**Kód třídy funkce:**

using System;

using System.Collections.Generic;

using System.Text;

namespace Semestralni\_konzolova\_aplikace

{

class Funkce

{

// vypocet taylorova clenu

private static double Tayloruv\_clen (double promenna, int poradiClenu) // vstupem je vstpní hodnota x a pořadí členu rozvoje

{ // výpočet čitatele členu taylorova rozvoje

double citatel = 1;

for (int i = 0; i < poradiClenu; i++)

{

citatel = citatel \* (promenna - 1);

}

// výpočet jmenovatele členu taylorova rozvoje

int jmenovatel = poradiClenu;

double clen = citatel / jmenovatel;

// znaménko členu (ale se každý lichý člen přičetl a každý sudý odečetl)

if (poradiClenu % 2 == 0)

clen = -clen;

return clen; // vrací hodnotu jednoho členu taylorova rozvoje

}

// vypocet taylorova rozvoje

public static double Tayloruv\_rozvoj(double promenna, int pocetClenu)

{

double rozvoj = 0;

for (int j = 0; j < pocetClenu; j++) // cyklus pro přičítání jednotlivých členu taylorova rozvoje

{

rozvoj = rozvoj + Tayloruv\_clen(promenna, j + 1); // funkce Tayloruv\_clen výše

}

return rozvoj; // vrátí výslednou hodnotu taylorova rozvoje vypočítaného na zadaný počet iterací

}

// prevraceni hodnoty pokud je hodnota menší než 1

public static double Prevraceni(double promenna)

{

if (promenna < 1) // pokud je hodnota menší než nula, převrátí se

{

promenna = 1 / promenna;

}

else { } // pokud je větší než 0, zůstne stejná

return promenna; // metoda vrací stejnou vstupní hodnotu, nebo převrácenou vstupní hodnotu

}

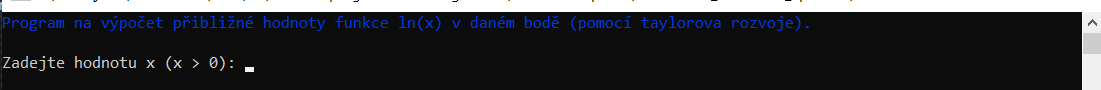
}

}

# Ukázka programu v Console Application

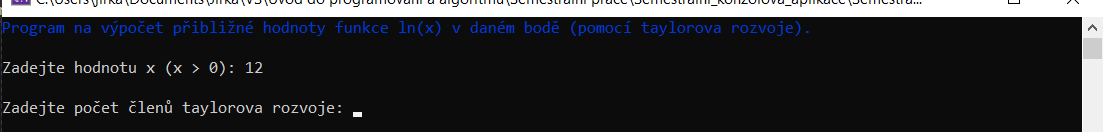
**Ukázka základních funkcí programu:**

Program nejprve vypíše text s informací o svojí funkci a vybídne uživatele k zadání prvního vstupu, kterým je hodnota pro výpočet (viz obr. 1):



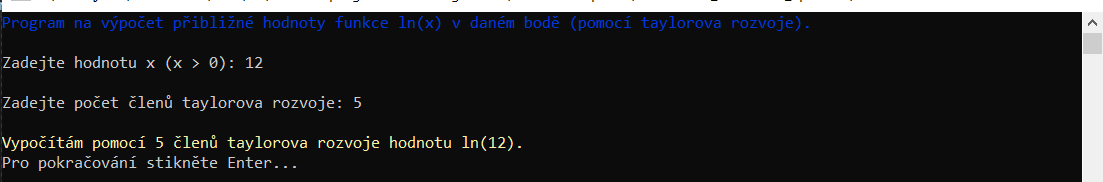
obr. - Vstup 1

Dále program vyžaduje zadání druhého uživatelského vstupu s přesností vypočtu (viz obr. 2):



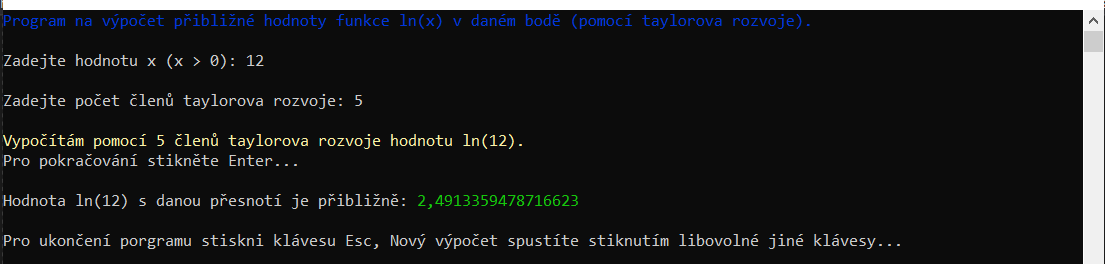
obr. - Vstup 2

Po zadání vstupů dojde k vypsání zadaných hodnot. Po překontrolování uživatel pokračuje stisknutím klávesy Enter (viz obr. 3):



obr. - Kontrola vstupu

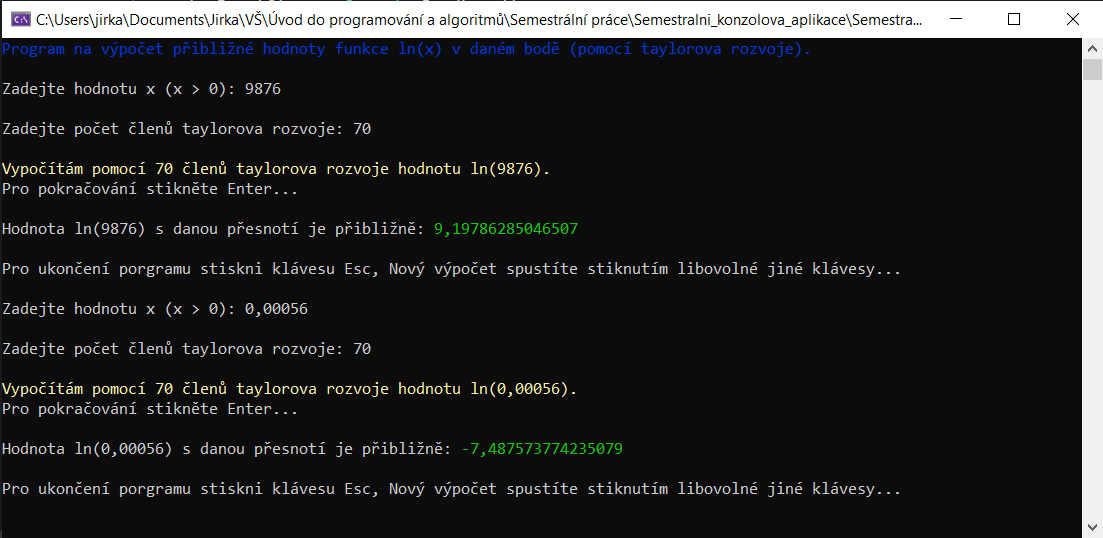
Nakonec se vypíše výsledek výpočtu zvýrazněný zelenou barvou. Uživatel má možnost program ukončit klávesou Esc nebo pokračovat dalším výpočtem (viz obr. 4):



obr. - Výsledek výpočtu

**Ukázka funkce programu při zadání velmi vysoké a velmi nízké hodnoty x:**

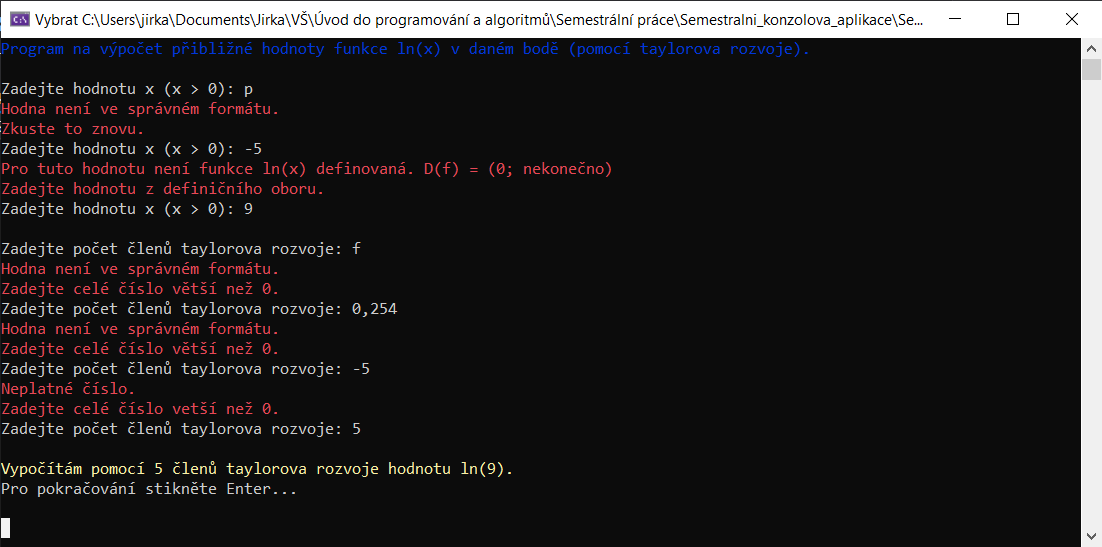
Program funguje při zadaní libovolně vysoké či nízké hodnoty v definičním obor přirozeného logaritmu (viz obr. 5):



obr. - Extrémní hodnoty

**Ukázka chybových hlášek při zadání nevhodných vstupů:**

Program má ošetřeny také chybné vstupy, po jejichž zadání se zobrazují chybové hlášky (viz obr. 6):



obr. - Chybové hlášky

# Závěr

Úkol pro mě byl zajímavou ukázkou toho, jak počítače vyhodnocují hodnoty složitějších funkcí, a následným zjištěním, že se nejedná o zdaleka tak složitý algoritmus, jaký jsem předpokládal. Po vypočítání několika členů Taylorova rozvoje logaritmu jsem odvodil jeho jednoduchost a naprogramování už šlo snadno. Nejvíce mě bavilo postupné zjednodušování funkce algoritmu i když místy jsem narážel zejména na syntaktické problémy. Všechny problémy jsem ale nakonec zdárně vyřešil.