



**VYSOKÁ ŠKOLA  
CHEMICKO-TECHNOLOGICKÁ  
V PRAZE**

# **Určení hodnoty Ludolfova čísla pomocí řady, kterou navrhl James Gregory**

Semestrální práce

Autor: Vladislav Aulich

Kruh: 159

Studijní obor: Fyzikální a výpočetní chemie

Fakulta: Fakulta chemicko-inženýrská

Akademický rok: 2021/2022

Předmět: Úvod do programování a algoritmů

Vedoucí práce: Ing. Jan Kohout

Praha, 2021

# Obsah

<b>1</b>	<b>Teoretická část</b>	<b>3</b>
1.1	Zadání úlohy . . . . .	3
1.2	Úvod . . . . .	3
<b>2</b>	<b>Implementace</b>	<b>4</b>
2.1	Postup řešení . . . . .	4
2.1.1	Časová složitost algoritmu . . . . .	4
2.2	Použitý software . . . . .	4
2.3	Zdrojový kód . . . . .	5
2.4	Ukázka programu . . . . .	6
	<b>Závěr</b>	<b>8</b>
	<b>Seznam použité literatury</b>	<b>9</b>
	<b>Seznam obrázků</b>	<b>10</b>

# 1. Teoretická část

## 1.1 Zadání úlohy

Určete hodnotu Ludolfova čísla pomocí řady, kterou navrhl James Gregory.

## 1.2 Úvod

Snahy o stanovení hodnoty čísla  $\pi$  sahají hluboko do historie. Jednou z mnoha možností je vyjádření Ludolfova čísla pomocí Maclaurinova rozvoje funkce  $\arctan(x)$ . Za objevitele této řady bývá považován skotský matematik James Gregory, který ji objevil v polovině 17.století. Nezávisle na Gregorym dospěl k stejné řadě také Gottfried W. Leibniz, proto také bývá někdy označována jako Leibnizova nebo Gregory-Leibniz řada.

$$\arctan(x) = x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \frac{x^9}{9} \dots$$

Dosazením  $x = 1$  získáme funkční hodnotu funkce  $\arctan(1) = \frac{\pi}{4}$ . Pokud tyto úpravy provedeme pro celou řadu, získáme pro aproximaci čísla  $\pi$  následující vztah:

$$\frac{\pi}{4} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} \dots$$
$$\pi = 4 \left( 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} \dots \right)$$

Tato řada konverguje ke skutečné hodnotě čísla  $\pi$  velmi pomalu, pro určení čísla s přesností na dvě desetinná místa musíme provést téměř 300 iterací<sup>1</sup>. Tato vlastnost je je citelná zejména pro ruční výpočty, které musel Gregory provádět.

Z této řady vychází další myšlenky pro stanovení čísla  $\pi$ . Zrychlení konvergence řady navrhl Abraham Sharp, který dosadil hodnotu  $x = \sqrt{\frac{1}{3}}$ . Tímto způsobem následně získal hodnotu čísla na 72 platných cifer.

---

<sup>1</sup><https://web.archive.org/web/20071128201346/http://www.scm.org.co/Articulos/832.pdf>

## 2. Implementace

### 2.1 Postup řešení

Pro součet této řady byla vytvořena metoda nazvaná `VypocitejPi`. Vstupní parametr je počet iterací (členů řady), pro které se má stanovit hodnota  $\pi$ . Parametr je datového typu `uint`, protože se jedná o celé kladné číslo. Návrátovou hodnotou je stanovený výsledek čísla  $\pi$  datového typu `decimal`.

V metodě je využito pomocné proměnné „pomocna“, která zaznamenává hodnotu zlomku. Následně je v programu rozhodovací logika, která buď hodnotu zlomku přičte nebo odečte od hodnoty proměnné „soucet“. Pro liché zlomky je znaménko kladné, pro sudé záporné.

Dále je v programu ošetřován uživatelský vstup „n“, který označuje počet iterací. Ten byl zadáním omezen na 1000, stejně jako zaokrouhlení výstupu na 8 desetinných míst. V programu je dále použita pomocná proměnná „pokracovat“, slouží k opakování načtení vstupu v případě špatně zadaného vstupu (do-while cyklus).

#### 2.1.1 Časová složitost algoritmu

Program obsahuje pouze jeden for cyklus, který se provádí  $n$  krát v závislosti na délce uživatelského vstupu. Algoritmus tedy spadá do časové složitosti  $\mathcal{O}(n)$ .

### 2.2 Použitý software

Pro tvorbu dokumentace bylo využito prostředí online  $\text{\LaTeX}$  editoru Overleaf. Jako podklad byla využita šablona k maturitní práci Gymnázia Jana Keplera, která byla graficky upravena.

Programování samotné aplikace probíhalo ve vývojovém prostředí Visual Studio 2017 od společnosti Microsoft. Změny na projektu byly průběžně zaznamenávány verzovacím systémem Git a následně vkládány na server Github. Odkaz na repozitář: [https://github.com/aulichv/pi\\_Gregory.git](https://github.com/aulichv/pi_Gregory.git).

## 2.3 Zdrojový kód

```
1      using System;
2      using System.Collections.Generic;
3      using System.Linq;
4      using System.Text;
5      using System.Threading.Tasks;
6
7      namespace semestralka_konsole
8      {
9          class Program
10         {
11             /// <summary>
12             /// Metoda, která pomocí rady Jamese Gregoryho
13             /// stanoví hodnotu Ludolfova čísla.
14             /// </summary>
15             /// <param name="n">Počet iterací (člen rady)</param>
16             /// <returns>Hodnota pi datového typu decimal</returns>
17             public static decimal VypocitejPi(uint n)
18             {
19                 decimal soucet = 0;
20                 decimal pomocna = 0;
21                 for (int i = 0; i < n; i++)
22                 {
23
24                     pomocna = 4m / (1 + (2 * i));
25                     //Pro sude koeficienty
26                     if (i % 2 == 0)
27                         soucet += pomocna;
28                     //Pro liche koeficienty
29                     else
30                         soucet -= pomocna;
31                 }
32                 return soucet;
33             }
34             static void Main(string[] args)
35             {
36                 bool pokračovat = true;
37                 uint n = 0;
38                 decimal pi = 0;
39                 do
40                 {
41                     Console.WriteLine("Zadejte počet iterací:");
42                     try
43                     {
44                         n = uint.Parse(Console.ReadLine());
45                         //Stanovení podmínky počtu opakování
46                         if (n > 1000)
```

```
47         {
48             //Upozorni uzivatele na spatny vstup
49             Console.WriteLine("Zadejte prosim cele kladne
               cislo mensi nez 1 000");
50         }
51         else
52             //Spravna hodnota n, ukonci smycku a vypocte pi
53             pokracovat = false;
54     }
55
56     catch (Exception e)
57     {
58         Console.WriteLine("Zadejte prosim cele kladne cislo
               mensi nez 1 000");
59     }
60 }
61 while (pokracovat);
62 //Provedeni funkce pro uzivatelem zadany pocet iteraci a
        zaokrouhleni na 8 desetinnych mist
63 pi = Math.Round(VypocitejPi(n), 8);
64 Console.WriteLine("Hodnota Ludolfova cisla pro {0} iteraci je
               {1}", n, pi);
65 Console.ReadKey();
66     }
67 }
68 }
```

Listing 2.1: Výsledný zdrojový kód

## 2.4 Ukázka programu



Obrázek 2.1: Obrazovka po spuštění programu

```
Zadejte počet iterací:  
h  
Zadejte prosím celé kladné číslo menší než 1 000  
Zadejte počet iterací:
```

Obrázek 2.2: Hláška programu při špatném uživatelském vstupu

```
Zadejte počet iterací:  
h  
Zadejte prosím celé kladné číslo menší než 1 000  
Zadejte počet iterací:  
200  
Hodnota Ludolfova čísla pro 200 iterací je 3,13659268
```

Obrázek 2.3: Výstup programu pro zadaný počet iterací

# Závěr

Při práci na projektu jsem se dozvěděl mnoho zajímavostí o historii čísla  $\pi$ . Zejména o jeho významu v matematice a snahách o stanovení jeho hodnoty.

Dále jsem rozšířil své znalosti o sázení zdrojového kódu v sázecím jazyce  $\text{\LaTeX}$ . Původně jsem kód vykresloval pomocí balíčku „verbatim“, u toho se mi však nepovedlo nastavit kód podle mých představ. Nakonec jsem tedy použil balíček „listings“. U něj jsem narážel na problémy s kódem obsahujícím diakritiku.

K tvorbě dokumentace jsem využíval webový nástroj Overleaf a jednotlivé verze jsem verzoval v repozitáři umístěném na serveru Github. Nové zkušenosti jsem získal i přizpůsobením si vývojového prostředí Visual Studio.

Práci na projektu považuji za přínosnou, domnívám se, že jsem zadání splnil.



# Seznam použité literatury

- [12] *How to change listing caption?* [online], [cit. 2021-11-13]. 2012. URL: <http://tilia.wikidot.com/citace-internetovych-zdroju-v-bibtexu>.
- [aut21] Tilia Cordata authors. *Citace internetových zdrojů v BibTeXu*. [online], [cit. 2021-11-12]. 2021. URL: <http://tilia.wikidot.com/citace-internetovych-zdroju-v-bibtexu>.
- [Ber21] Eliška Bernátová. “Historie čísla  $\pi$ ”. [Online; cit. 12. 11. 2021]. Dipl. pr. Univerzita Karlova, 2021. URL: <http://hdl.handle.net/20.500.11956/51152>.
- [FRA13] Jan FRANK. *Číslo pi a jeho aproximace* [online]. Bakalářská práce. 2014 [cit. 2021-11-13]. URL: [Dostupn%C3%A9%20z%20WWW%20%3Chttps://theses.cz/id/ysaypx/%3E](https://theses.cz/id/ysaypx/%3E).
- [Mic21] Microsoft. *Dokumentace k jazyku C#*. [online], [cit. 2021-11-12]. 2021. URL: <https://docs.microsoft.com/cs-cz/dotnet/csharp/>.
- [Ove21] Overleaf. *Code listing*. [Online; cit. 12. 11. 2021]. 2021. URL: [https://cs.overleaf.com/learn/latex/Code\\_listing](https://cs.overleaf.com/learn/latex/Code_listing).
- [Wik21] Wikipedie. *Pí (číslo)* — *Wikipedie: Otevřená encyklopedie*. [Online; cit. 12. 11. 2021]. 2021. URL: [https://cs.wikipedia.org/w/index.php?title=P%C3%AD\\_\(%C4%8D%C3%ADslo\)&oldid=20398279](https://cs.wikipedia.org/w/index.php?title=P%C3%AD_(%C4%8D%C3%ADslo)&oldid=20398279).

# Seznam obrázků

2.1	Obrazovka po spuštění programu . . . . .	6
2.2	Hláška programu při špatném uživatelském vstupu . . . . .	7
2.3	Výstup programu pro zadaný počet iterací . . . . .	7

# Seznam kódů

2.1	Výsledný zdrojový kód . . . . .	5
-----	---------------------------------	---