

Tomáš Musílek
158

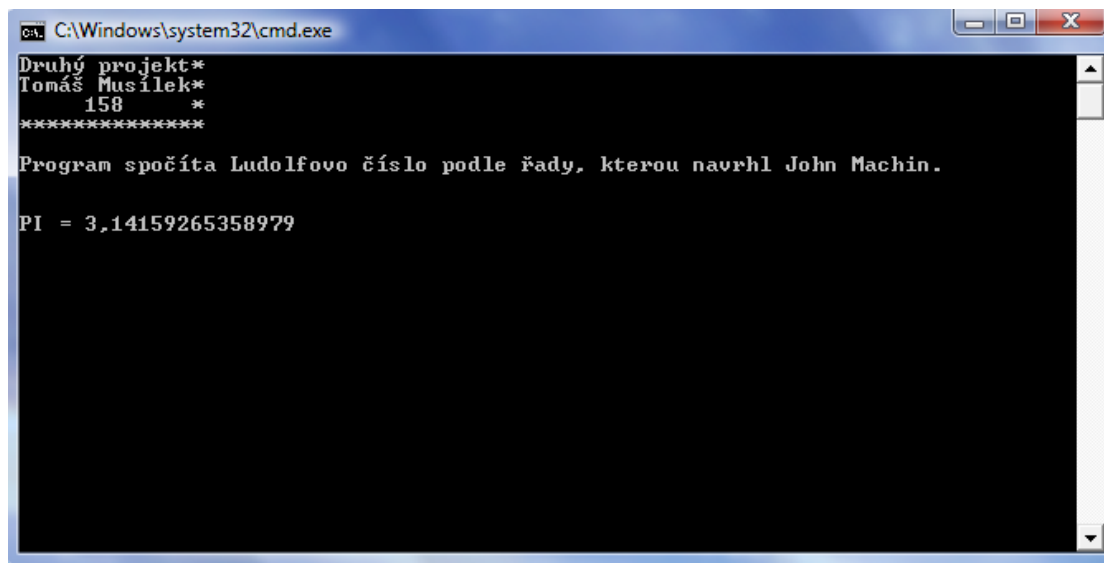
Druhý projekt

Zadání: Určete hodnotu Ludolfova čísla pomocí řady, kterou navrhl John Machin.

Vzorečky:
$$\frac{\pi}{4} = 4 \operatorname{arctg}\left(\frac{1}{5}\right) - \operatorname{arctg}\left(\frac{1}{239}\right) = 4 \sum_{k=0}^{\infty} \frac{(-1)^k}{(2k+1)5^{2k+1}} - \sum_{k=0}^{\infty} \frac{(-1)^k}{(2k+1)239^{2k+1}}$$

Popis řešení: Na základě předpisu, jsem napsal program, který už po devíti iteracích dosáhne takové přesnosti, že číslem π se naplní celý rozsah platných číslic double, což je asi 15 číslic. Navrhl jsem výpočet pomocí dvou metod. 1. metoda vychází přímo z předpisu, pro výpočet π , druhá metoda pro každou iteraci nedopočítává už všechny hodnoty od začátku, ale používá hodnot předchozích.

Program vypíše tento výpis:



```
C:\Windows\system32\cmd.exe
Druhý projekt*
Tomáš Musilek*
158
*****
Program spočítá Ludolfovo číslo podle řady, kterou navrhl John Machin.
PI = 3,14159265358979
```

Výsledkem programu je číslo π s přesností na 14 desetinných míst. Vypočítat toto číslo s větší přesností nám už double neumožňuje. Protože 14 desetinných míst se mi zdálo málo, rozhodl jsem se tento program modifikovat tak, aby byl schopen větší přesnosti. Výsledkem je program s názvem Projekt2.1, který je součástí přílohy. Tento program pro výpočet nepoužívá double, ale pole bytů, což mu dovolí číslo π spočítat na mnohem víc desetinných míst. To je samozřejmě spojeno s větší časovou náročností.

Ukázka z výpisu programu Projekt2.1:

K výpočtu byla použita řídicí proměnná k jdoucí od nuly do 999. Z pokusů jsem usoudil, že zvýšením k o jedničku a zařazením tohoto k do výpočtu, stoupne přesnost výpočtu o něco více než o jedno desetinné místo. Na konci programu jsou vypočítána dvě čísla π , přičemž maximální k , z něhož byla vypočítána, se liší u obou výsledků pouze o jedničku. Na obrazovku je vypsáno jen tolik desetinných míst, kolik mají obě čísla společná. Což v tomto případě dává 1400 desetinných míst čísla π . Tento výpočet trval přibližně 22 minut.

```
C:\Windows\system32\cmd.exe
3.141592653589793238462643383279502884197169399375105820974944592307816406286208
99862803482534211706798214808651328230664709384460955058223172535940812848111745
02841027019385211055596446229489549303819644288109756659334461284756482337867831
65271201909145648566923460348610454326648213393607260249141273724587006606315588
17488152092096282925409171536436789259036001133053054882046652138414695194151160
94330572703657595919530921861173819326117931051185480744623799627495673518857527
24891227938183011949129833673362440656643086021394946395224737190702179860943702
77053921717629317675238467481846766940513200056812714526356082778577134275778960
91736371787214684409012249534301465495853710507922796892589235420199561121290219
60864034418159813629774771309960518707211349999998372978049951059731732816096318
59502445945534690830264252230825334468503526193118817101000313783875288658753320
83814206171776691473035982534904287554687311595628638823537875937519577818577805
32171226806613001927876611195909216420198938095257201065485863278865936153381827
96823030195203530185296899577362259941389124972177528347913151557485724245415069
59508295331168617278558890750983817546374649393192550604009277016711390098488240
12858361603563707660104710181942955596198946767837449448255379774726847104047534
64620804668425906949129331367702898915210475216205696602405803815019351125338243
003558764024749647326391419927260426992279
00:21:55.0820000
*Pokračujte stisknutím libovolné klávesy...
```

Závěr: Výpočet čísla π pomocí double, nedostaneme sice takovou přesnost, ale s ohledem na časovou náročnost tvoření tohoto kódu a kódu pro výpočet pomocí pole bytů, je tato metoda zcela postačující. V praxi stejně nebudeme nikdy potřebovat více než 11 desetinných míst. Program, který počítá π pomocí pole bytů, by šel ještě výrazně zrychlit, například tím, že by se zavedli proměnné pro uložení hodnot mezivýpočtu, poté by se nemuselo např. $5^{(2k+1)}$ počítat pořád od začátku, ale stačilo by předchozí hodnotu vynásobit 25.

Zdroje: Wikipedia.org