Univerzita Karlova

Přírodovědecká fakulta

Úvod do programování

Iterativní výpočet odmocniny z čísla *c* s chybou menší než *ε*

Zkouška

Adam Bartůšek 3. ročník BGEKA 10.02.2022

1. ZADÁNÍ

Mějme na vstupu reálné nezáporné číslo c a reálné nenulové číslo ε. Úkolem programu je iterativně spočítat odmocninu (sqrt(c)) z čísla c s chybou menší než ε. Toho má být dosaženo pouze za pomoci základních matematických operací, tedy sčítání, odčítání, násobení a dělení.

2. POPIS A ROZBOR PROBLÉMU + VZORCE

Problém iterativního výpočtu odmocniny z čísla lze řešit více způsoby. V této práci byla pro svou jednoduchost zvolena tzv. Babylonská metoda výpočtu odmocniny. Postup je následující.: Chceme zjistit odmocninu z reálného nezáporného čísla c, vytvoříme tedy první odhad odmocniny, čím bližší je skutečné odmocnině, tím méně následujících výpočtů je potřeba. Následně aktualizujeme náš první odhad, a to pomocí vzorce:

$x_{n+1} = (x_n + c/x_n)/2,$

kde x_{n+1} je nový odhad, c je číslo, jehož odmocninu chceme vypočítat a x_n je odhad předchozí.

Tento krok se opakuje až do doby, než je rozdíl dvou po sobě jdoucích odhadů menší než předem stanovená chyba. Formálně lze ukončení procesu popsat jako krok, kdy platí

$|x_n - x_{n+1}| < \varepsilon,$

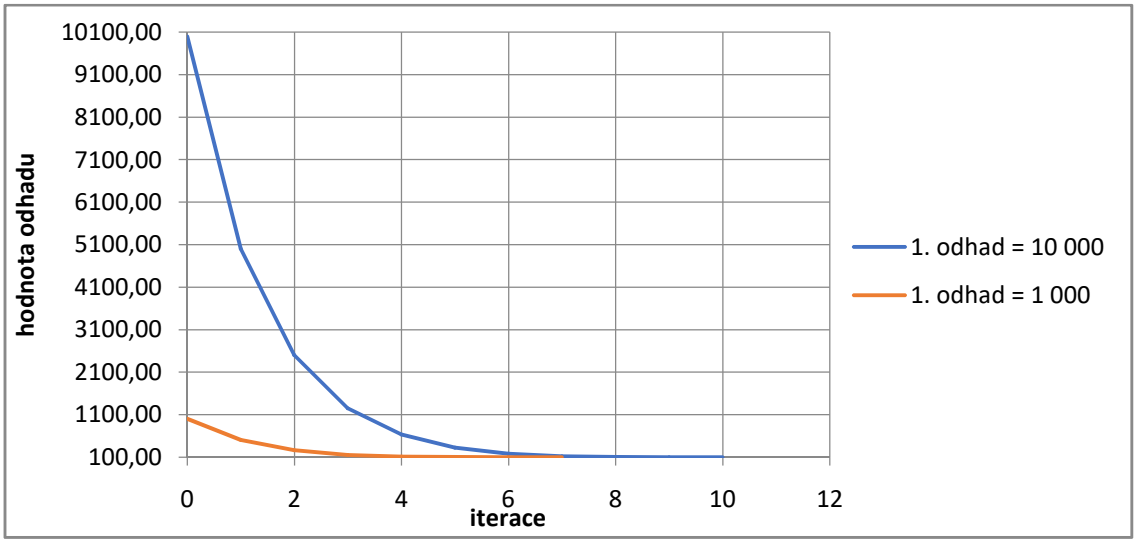
kde ε je stanovená chyba.

3. POPISY ALGORITMŮ FORMÁLNÍM JAZYKEM

Tato metoda vychází ze základní myšlenky, že pokud je odhad x_0 odmocniny z čísla c větší než skutečná odmocnina, pak c/x_0 je menší nežli skutečná odmocnina a vice versa. Průměr x_0 a c/x_0 (což je nový, následující odhad – viz část 2) se pak více blíží skutečné odmocnině. Každý vypočítaný (tedy ne 1.) odhad je větší než skutečná odmocnina – jinými slovy geometrický průměr (v tomto případě se geometrický průměr rovná odmocnině)

$\text{sqrt}(\text{est} * c/\text{est}) = \text{sqrt}(c) \quad \leq \quad (\text{est} + c/\text{est})/2$

je vždy menší nebo rovný aritmetickému průměru. Každý další vypočítaný odhad x_{n+1} tedy konverguje se skutečnou odmocninou a je nižší než předchozí odhad x_n (výjimkou mohou být odhady x_0 a x_1, pokud první odhad x_0 je menší než skutečná odmocnina). Tato konvergence je pro dva různé počáteční odhady na odmocninu z 10 000 znázorněna na obr. 1.



Obr. 1: Konvergence odhadů odmocniny z 10 000 s přibývajícími iteracemi

4. PROBLEMATICKÉ SITUACE A JEJICH ROZBOR + JEJICH OŠETŘENÍ V KÓDU

První problematickou situací je již samotný výběr prvního odhadu. Správně vybraným prvním odhadem je docíleno snížení počtu kroků, nicméně výpočet je tak jednoduchý, že počítač iterativně vypočítá odmocninu i z gigantických čísel téměř okamžitě – proto bylo jako 1. odhad v programu vybráno číslo stejné jako původní číslo – to proto, aby se předešlo problematickým situacím při odmocňování malých čísel. Následně ale bylo upraveno jako (c+1)/2, jelikož tento odhad následuje vždy po prvním výpočtu (druhého) odhadu (pokud byl první odhad stejný jako odmocňované číslo), ať je číslo c jakékoli (reálné nezáporné).

Další problematickou situací je nekorektní vstup, tedy zadání záporného čísla c nebo nekladného čísla ε. Zadání záporného čísla c je nekorektní, jelikož v oboru reálných čísel nelze vypočítat odmocninu ze záporného čísla. Zadání záporného čísla ε samo o sobě nekorektní není, ale v kontextu výpočtu nedává smysl a proto je tato situace ošetřena vynásobením záporné chyby číslem -1. Zadání nulového ε nekorektní je, jelikož by byl výpočet nekonečný (iterativní metoda nikdy nevypočítá úplně přesnou hodnotu odmocniny). Když nastane jedna ze 2 předešlých situací, funkce vypíše příslušnou chybovou hlášku a vrátí hodnotu *None*.

Problematická situace nastane také, když je číslo c rovné 0. Sice by program vrátil hodnotu blízkou 0 (v závislosti na udaném ε), ale jelikož to je očekávaná, ojedinělá a snadno řešitelná situace (je známo, že sqrt(0) = 0), je to v kódu vyřešeno jednoduchým vrácením hodnoty 0. Uznávám ale, že toto ošetření není nutné, zadání by bylo splněno i bez něj. Problémem by to bylo v případě, že by první odhad byl roven číslu c, pak by došlo k dělení nulou.

5. VSTUPNÍ DATA, FORMÁT VSTUPNÍCH DAT, POPIS

Vstupními daty jsou pouze dvě čísla typu *float* - reálné nezáporné číslo c (odmocňované číslo) a reálné nenulové číslo ε (daná chyba).

6. VÝSTUPNÍ DATA, FORMÁT VÝSTUPNÍCH DAT, POPIS

Výstupními daty je vždy jedno nezáporné číslo typu *float*. Pokud nebyly dodrženy podmínky korektního vstupu, výstupní hodnota = *None*.

7. DOKUMENTACE

Program se sestává z jedné funkce *squareroot*. Ta má dva argumenty – *number* a *error*. *Number* reprezentuje číslo c, jenž má být odmocněno a *error* stanovenou chybu ε (tj. jak moc se může lišit skutečná odmocnina od té iterativně vypočítané). Následuje vytvoření prvního odhadu *est*, který je roven polovině součtu *number* a 1. Dále jsou ošetřeny výše zmíněné situace, kdy je číslo c záporné (vrátí se *None*), nulové (vrátí se 0) a když je číslo ε záporné či nulové (vrátí se *None*).

Následuje *While* cyklus, který v případě, že *number* je větší než 0, počítá pomocí vzorce *est_new = (est+(number/est))/2*, kde *est_new* je nový odhad a *est* je předchozí odhad, hodnotu nového odhadu. Na konci každého cyklu je přiřazena do proměnné *est* hodnota nového odhadu *est_new* a znovu se počítá nový odhad. V případě, že rozdíl původního a nového odhadu je menší než daná hodnota *error*, je cyklus ukončen a je vrácena hodnota současného (nového) odhadu.

Po definici funkce *squareroot* následuje její volání.

8. ZÁVĚR, MOŽNÉ ČI NEŘEŠENÉ PROBLÉMY, NÁMĚTY NA VYLEPŠENÍ

Tento program slouží k iterativnímu výpočtu odmocniny z čísla c s chybou ε za pomoci základních matematických operací tzv. Babylonskou metodou. Program umí spočítat odmocninu z přiměřeně velkých reálných nezáporných čísel téměř okamžitě. Neumí však např. spočítat odmocninu z komplexních čísel. V současné době mě nenapadá žádné vylepšení, dalo by se např. použít jiné metody (Newtonova metoda apod.), na funkčnost programu by to ale nemělo mít vliv.

9. SEZNAM LITERATURY

CECHA, M., 2019: Nízkoúrovňové numerické výpočty. Bakalářská práce. Brno, 2019. https://is.muni.cz/th/gezhn/bakfina.pdf