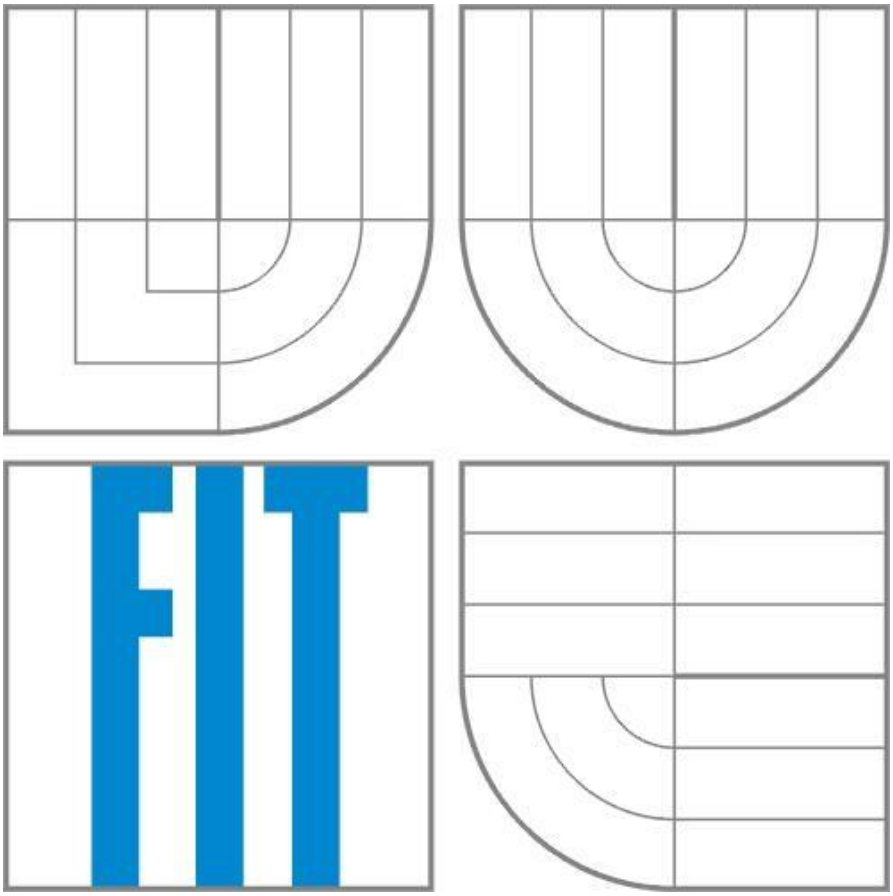


DOKUMENTÁCIA K PROJEKTU PRE PREDMETY IZP A IUS

Iteračné výpočty

2.projekt

5. december 2014

Autor: Maroš Vasilišin, xvasil02@stud.fit.vutbr.cz

Fakulta Informačných Technológií

Vysoké učení technické v Brne

Obsah

1 Úvod

Tento dokument detailne popisuje návrh a implementáciu riešenia pre výpočet tangensu dvomi rôznymi iteračnými metódami: pomocou Taylorovho polynómu a pomocou zreťazených zlomkov. Takisto popisuje vlastné odvodenie počtu iterácií pre metódu zreťazených zlomkov. Oba algoritmy sú použité pre výpočet vzdialenosti a výšky meraného objektu meracím prístrojom podľa zadania 2. projektu predmetu IZP.

Program funguje vo forme konzolovej aplikácie, ktorá očakáva hodnoty na štandardnom vstupe a na štandardný výstup vypíše výsledok. Hodnoty musia byť vo formáte špecifikovanom v zadaní.

Dokument sa skladá z viacerých častí: v kapitole 2 sa budem zaoberať analýzou zadaného problému, kapitola 3 sa venuje návrhu možných riešení a v kapitole 5 sa už budem venovať konkrétnemu riešeniu a jeho implementácií.

Celý projekt je napísaný v programovacom jazyku C.

2 Analýza problému a princíp jeho riešenia

Základným predpokladom pre úspešné vytvorenie vlastnej funkcie tangens je, že táto funkcia sa dá napísať ako postupnosť jednoduchých matematických operácií. Pre výpočet tangensu sú to dve rôzne metódy: Taylorov rozvoj a metóda zreťazených zlomkov. V nasledujúcej časti sa budem zaoberať oboma metódami a načrtnem princíp ich riešenia.

2.1 Zadanie problému

Princípom projektu je vytvorenie programu v programovacom jazyku C, ktorý zo zadaných vstupných údajov vypočíta tangens daného uhla na daný počet iterácií, porovná výsledky výpočtov obidvoch implementovaných metód výpočtu s funkciou **tan()** z matematickej knižnice **math.h** a na štandardný výstup vypíše výsledky všetkých výpočtov a ich absolútne odchýlky. Pre iné vstupné argumenty vypočíta pomocou metódy zreťazených zlomkov vzdialenosť a výšku meraného objektu. V zadaní je špecifikované, že uhol musí byť väčší ako **0** a maximálne **1.4** radiánov. Pre metódu Taylorovho polynómu sa uvažuje presnosť na **13** prvých členov polynómu. Pre metódu zreťazených zlomkov sa počet zanorení odvodí pomocou funkcie na výpočet presnosti na **10 miest**.

2.2 Presnosť výpočtu

Pre zadaný projekt je presnosť výpočtu obmedzená už v zadaní. Pre výpočet metódou Taylorovho polynómu je výpočet obmedzený na prvých 13 členov polynómu a teda skutočný tangens sa bude trocha líšiť od nami vypočítaného. Pre nás ale výpočet na viac členov nemá zmysel, pretože sa výsledky budú líšiť až na nedôležitých desatinných miestach.
Pre výpočet metódou zreťazených zlomkov bolo potrebné odvodiť počet zanorení zreťazeného zlomku. V zadaní bola špecifikovaná presnosť na 10 miest. Tú dosiahneme tým, že porovnávame práve kontrolovaný člen zlomku s predchádzajúcim, a ak sa ich rozdiel líši až na 11. mieste, dosiahli sme požadovanú presnosť na 10 miest. Týmto spôsobom teda vieme odvodiť počet iterácií potrebných na dosiahnutie požadovanej presnosti. Podrobnejší popis funkcie, ktorá to počíta je uvedený v kapitole 5.4.

2.3 Taylorov polynóm

Taylorov polynóm sa v matematike využíva na odvodenie elementárnych, ale aj zložitejších funkcií. Ako príklad uvediem exponenciálnu funkciu e^x, sínus, kosínus, tangens, logaritmus alebo nekonečný rad. Na výpočet s dokonalou presnosťou by bolo potrebné vypočítať nekonečno členov, pre k-členov sa teda musíme uspokojiť s približnou presnosťou.
Pre výpočet tangensu jeho začiatok vyzerá takto:

$$\tan(x) = x + \frac{x^3}{3} + \frac{2x^5}{15} + \frac{17x^7}{315} + \frac{62x^9}{2835} + \cdots$$

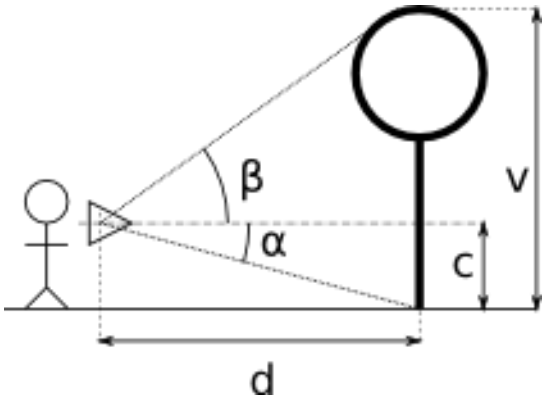
2.4 Zreťazený zlomok

V matematike je zreťazený zlomok výraz, ktorý je zapísaný ako súčet čísla a rekurziou vypočítaného ďalšieho čísla, ktoré je znova súčet čísla a rekurziou vypočítaného čísla atď. Ak poznáme presný počet zanorení zlomku, hovoríme o konečnom zreťazenom zlomku, ak počet zanorení nepoznáme, ide o nekonečný zreťazený zlomok.
Základný tvar zreťazeného zlomku vyzerá takto:

$$a_0 + \cfrac{b_1}{a_1 + \cfrac{b_2}{a_2 + \cfrac{b_3}{a_3 + \cdots}}}$$

2.5 Výpočet vzdialenosti a výšky

Pre výpočet vzdialenosti a výšky objektu, potrebujeme poznať uhly.
V 2. projekte sme mali zadaný tento obrázok:



d – vzdialenosť meracieho prístroja od meraného objektu
v – výška meraného objektu
c – výška meracieho prístroja

Pre výpočty potrebujeme poznať vzorec pre tangens v trojuholníku:

$$\tan(\alpha) = \frac{protilahlá\ odvesna}{prilahlá\ odvesna}$$

3 Návrh riešenia problému

Pri riešení problému budem vychádzať hlavne z analýzy. Pri návrhu riešenia je dôležité zvoliť si správne dátové typy výsledkov. Keďže budeme pracovať s desatinnými číslami, najvhodnejšie je použiť dátový typ **double**, čo vyplýva aj zo zadania úlohy. Pre počet iterácií je vhodné použiť dátový typ **int**. Argumenty sa budú načítavať zo štandardného vstupu a výsledky sa budú zapisovať pomocou príkazu printf na štandardný výstup vo formáte %e pre funkciu tangens a vo formáte %.10e pre výpočet vzdialenosti a výšky. Ukladanie výsledkov nie je podľa zadania potrebné.

3.1 Tangens

Pre výpočet tangensu pomocou Taylorovho polynómu potrebujeme poznať prvých 13 koeficientov čitateľa a menovateľa. Pre daný definičný obor teda tangens vypočítame postupným dosadzovaním jednotlivých koeficientov do vzorca(viď kapitola 2.3) a zvyšovaním koeficientu pri x. Jednotlivé koeficienty boli uvedené v zadaní úlohy.

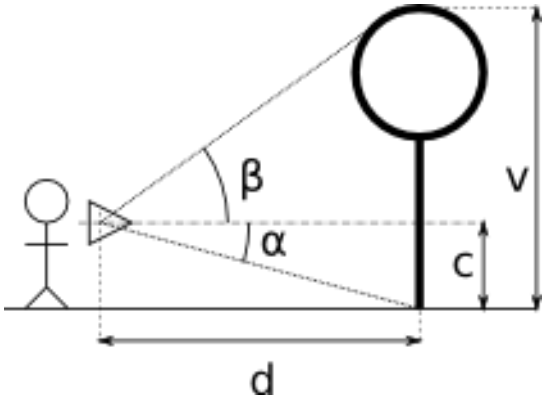
Pre výpočet tangensu pomocou metódy zreťazených zlomkov potrebujeme poznať presnú špecifikáciu zlomku(viď kapitola 2.4) pre tangens. Zreťazený zlomok pre tangens má tvar:

$$\tan(x) = \cfrac{x}{1 - \cfrac{x^2}{3 - \cfrac{x^2}{5 - \cfrac{x^2}{7 - \dots}}}}$$

Počet zanorení pre zadanú presnosť na 10 miest odvodíme tak, že si určíme najprv počet zanorení na 1 a budeme ho zvyšovať. Budeme testovať či sme už nedosiahli požadovanú presnosť tak, že porovnáme posledné dva členy zreťazeného zlomku a ak ich rozdiel je menší ako $1.0e^{-10}$, tak sme danú presnosť dosiahli. Ak nie je, tak pokračujeme ďalej v cykle.

3.2 Výpočet vzdialenosti a výšky

Výpočet vzdialenosti a výšky budem realizovať pomocou výpočtu tangensu metódou zreťazených zlomkov(viď kapitola 3.1) a dosadením do vzorca pre tangens uhla v trojuholníku(viď kapitola 2.5).



d – vzdialenosť meracieho prístroja od meraného objektu
v – výška meraného objektu
c – výška meracieho prístroja

Ak chceme vypočítať vzdialenosť (d), tak potrebujeme vypočítať tangens uhla α(alfa). Podľa zadania máme použiť nami vytvorenú funkciu pre výpočet tangensu metódou zreťazených zlomkov.

Vzorec bude **d = c / tan(α)**

Pre výpočet výšky (v) potrebujeme poznať tangens uhla β(beta).
Podľa zadania aj tu máme využiť nami vytvorenú funkciu pre výpočet pomocou zreťazených zlomkov.

Vzorec bude **v = c + (d*tan(β))**

4 Špecifikácia testov

Pri používaní programu môže nastať mnoho situácií, keď používateľ zadá nesprávne argumenty. Preto treba testovať všetky možnosti. V tomto projekte sú všetky rozsahy pevne dané, takisto počet argumentov je stály. Ak používateľ zadá niečo nesprávne, program by mal vyhodiť chybové hlásenie.

1. Test: žiaden argument

Príkazový riadok:	Očakávaný výstup:	Výstup:
./proj2	Chyba	Chyba

2. Test: chybná syntax

Príkazový riadok:	Očakávaný výstup:	Výstup:
./proj2 --helxp	Chyba	Chyba

3. Test: chybný počet argumentov

Príkazový riadok:	Očakávaný výstup:	Výstup:
./proj2 --tan 1.12 6 8 9	Chyba	Chyba

4. Test: chybný rozsah

Príkazový riadok:	Očakávaný výstup:	Výstup:
./proj2 --tan 0 6 10	Chyba	Chyba
./proj2 --tan 1.8 6 10	Chyba	Chyba
./proj2 --tan 1.12 1 18	Chyba	Chyba

5. Test: argument nie je číslo

Príkazový riadok:	Očakávaný výstup:	Výstup:
./proj2 --tan 1.1x2 6 10	Chyba	Chyba
./proj2 --tan 1.12 6x 10	Chyba	Chyba
./proj2 --tan 1.12 1 1x	Chyba	Chyba
./proj2 -c 45x -m 0.3 0.8	Chyba	Chyba

6. Test: správny výpočet

Príkazový riadok:	Očakávaný výstup:	Výstup:
./proj2 --help	výpis nápovedy	výpis nápovedy
./proj2 -m 0.3 0.9	4.84909221556e+00	4.84909221556e+00
	7.6106234032e+00	7.6106234032e+00
./proj2 -c 1.7 -m 0.15 1.3	1.1248205560e+01	1.1248205560e+01
	4.2217188781e+01	4.2217188781e+01

./proj2 --tan 1.024 6 7
Očakávaný výstup:
6 1.642829e+00 1.634327e+00 8.502803e-03 1.642829e+00 3.298801e-09
7 1.642829e+00 1.639216e+00 3.613451e-03 1.642829e+00 1.794520e-11
Výstup: OK
6 1.642829e+00 1.634327e+00 8.502803e-03 1.642829e+00 3.298801e-09
7 1.642829e+00 1.639216e+00 3.613451e-03 1.642829e+00 1.794520e-11

5 Popis riešenia

V nasledujúcej časti popíšem moju vlastnú implementáciu jednotlivých funkcií v programovacom jazyku C. Pri implementácií som vychádzal zo záverov z analýzy a návrh riešenia.

5.1 Ovládanie programu

Argumenty príkazového riadku v programe špecifikujú funkciu, ktorú chce používateľ vykonať.
Možnosti sú nasledovné:

- **--help**
- **--tan A N M**
- **[-c X] -m A [B]**

Prepínač **--help** vytlačí nápovedu k programu.
Prepínač **--tan** špecifikuje, že chceme vypísať porovnanie hodnôt tangensu vypočítaného metódou Taylorovho polynómu a metódou zreťazeného zlomku s funkciou tan() z matematickej knižnice.
Prepínačom **-m** dávame programu najavo, že chceme počítať vzdialenosť a výšku objektu.
Argument **A** označuje uhol, ktorého tangens počítame.
Argumenty **N,M** označujú poradie iterácií, ktoré chceme vypísať.
Voliteľný argument **B** označuje uhol potrebný pre výpočet výšky objektu.
Voliteľná dvojica argumentov **-c X** mení výšku meracieho prístroja.

5.2 Použité dátové typy

O použitých dátových typoch som už písal v kapitole 3. V tejto kapitole vysvetlím, prečo som si zvolil práve dátový typ **double.** Hlavným dôvodom bolo, že tento dátový typ bol uvedený v zadaní ako povinný, ale prečo vlastne?

Prvým dôvodom je, že typ double poskytuje presnosť na 15 desatinných miest a v našom projekte potrebujeme len presnosť na 10 miest.
Pre reálne čísla ale existujú aj iné dátové typy.
Dátový typ **float** je narozdiel od double menej presný a typ **long double** zaberá zbytočne veľa miesta.
Pre obmedzenia tohto projektu preto úplne stačí typ double.

5.3 Implementácia programu

Program sa spúšťa s jedinou funkciou v maine. Je to funkcia **kontrola_arg**, ktorá skontroluje správnosť argumentov príkazového riadku, skontroluje tiež či sú správne rozsahy jednotlivých argumentov, a tiež ich syntax. Podľa toho, aké argumenty boli zadané, táto funkcia zavolá správnu funkciu. Ak boli zadané argumenty nesprávne, funkcia vypíše chybu na štandardný chybový výstup a skončí program. Funkcia je implementovaná ako postupnosť if-podmienok, každá možnosť zadania argumentov má vlastnú podmienku a podľa jej pravdivosti sa volajú jednotlivé podprogramy.

5.4 Implementácia tangensu (--tan)

Výpočet tangensu je v projekte vo dvoch funkciách.

Funkcia **taylor_tan** počíta tangens metódou Taylorovho polynómu. Funkcia má dva parametre, uhol, ktorého tangens počíta a počet členov polynómu(teda počet iterácií). Uhol je v rozsahu (0,1.4> radiánov. Počet iterácií je nastavený na maximálne 13. Vo funkcií sú v dvoch poliach uložené koeficienty čitateľov a menovateľov polynómu. Prebieha tam for-cyklus od prvého členu polynómu až po člen zadaný v parametroch funkcie. Výsledok sa priebežne ukladá do lokálnej premennej a zvyšuje sa exponent uhla.

Funkcia **cfrac_tan** počíta tangens metódou zreťazeného zlomku. Funkcia má rovnako ako funkcia taylor_tan dva parametre. Druhý parameter funkcie sa líši pre prepínače --tan a pre -m. Pre --tan je nastavený podobne ako pri taylor_tan na maximálne 13 iterácií. Ako je nastavený pre -m je popísané v kapitole 5.5. Funkcia je implementovaná ako while cyklus, ktorý prechádza zreťazený zlomok smerom zvnútra von. Koeficient je odvodený od počtu zanorení a výsledky sú rekurzívne viazané na predchádzajúci výsledok. Pre najvyššie zanorenie je vložený špeciálny while cyklus s iným čitateľom, pretože v najvyššom zanorení sa nevyskytuje x^2, ale iba x. Funkcia vracia výsledok typu double.

Pre porovnanie vypočítaných hodnôt s funkciou tan() z matematickej knižnice potrebujeme poznať absolútnu chybu výpočtu. Keďže je zakázaná funkcia fabs() z math.h, vytvoril som si vlastnú funkciu na výpočet absolútnej hodnoty. Nazval som ju **my_fabs**. Funguje s dvoma parametrami a to tak, že ich od seba odčíta, a ak je výsledok kladný, vráti ho, a ak je záporný, vráti jeho prevrátenú hodnotu.

Vypočítané hodnoty sa vypisujú na štandardný výstup v poradí:
poradie iterácie, funkcia tan(), funkcia taylor_tan, absolútna chyba taylor_tan, cfrac_tan, absolútna chyba cfrac_tan. Pre výpis hodnôt je vytvorená ďalšia funkcia **my_tan.**

5.5 Implementácia výpočtu vzdialenosti a výšky (-m)

Pre výpočet vzdialenosti a výšky sa používa funkcia cfrac_tan, ktorá je detailne popísaná v kapitole 5.4. Je tu však rozdiel v tom, že počet iterácií pre prepínač -m sa odvádza v samostatnej funkcií a nie je nastavený na max 13 ako tomu bolo v predchádzajúcom prípade.

Funkcia pre výpočet počtu iterácií sa nazýva **pocet_iteracii** a volá sa s jedným parametrom, ktorým je uhol, ktorého tangens počítame. Žiadaná presnosť je podľa zadania 1.0e-10, ktorá je uložená v makre PRESNOST. Funkcia funguje ako cyklus do-while. Porovnáva dva výsledky funkcie cfrac_tan dovtedy kým nie je dosiahnutá požadovaná presnosť. Tieto dva výsledky sú v podstate volania funkcie cfrac_tan s dvoma rôznymi úrovňami zanorenia, jedna je o 1 hlbšia ako druhá. Cyklus začína na úrovniach zanorenia 1 a 2 a vždy sa zvyšuje o 1. Ak funkcia dosiahne požadovanú presnosť, vracia počet iterácií, ktorý bol potrebný na jej dosiahnutie.

Ak máme teda vypočítaný tangens, vzdialenosť a výšku počítame podľa vzorca pre tangens v trojuholníku, ktorý je uvedený v kapitole 3.2. Pre tieto výpočty sa v programe nachádzajú ďalšie dve funkcie. Jedna z nich, **my_merA**, počíta vzdialenosť objektu od meracieho prístroja. Ak je zadaný aj druhý argument prepínača -m, B, tak sa volá funkcia **my_merAB** , ktorá počíta vzdialenosť aj výšku objektu.

V argumentoch sa môže nachádzať aj prepínač -c X, ktorý označuje výšku meracieho prístroja. Ak sa v argumentoch programu tento prepínač nenachádza, výška meracieho prístroja je implicitne nastavená v makre VYSKA na 1.5. Argument X je podľa zadania možné zadať v rozmedzí 0 až 100 metrov.

6 Záver

Program je rozdelený na podprogramy, ktoré pracujú efektívne a rýchlo. Program dosahuje očakávané výsledky pre všetky možnosti zadania argumentov. Program bol testovaný na operačných systémoch Windows 8.1 (64 bit) a Linux (32,64 bit). Program a jeho jednotlivé podprogramy sú ľahko prenositeľné na iné platformy, pretože sú písané normou C99 a takisto sú dôkladne komentované pre ich ďalšie editovanie.

7 Metriky kódu

Počet súborov: 1

Počet funkcií: 10

Počet riadkov kódu: 311

Veľkosť statických dát: 648 B

Veľkosť kódu: 11 673 B

Veľkosť spustiteľného súboru: 13 224 B (Windows 8.1 64 bit)