Laboratorium 1, zadanie 3

Badane funkcje oraz argumenty

- f(x) = sqrt(x ^ 2 + 1) – 1
- g(x) = x^2/ (sqrt(x^2 + 1) + 1)
- x = 8^-1, 8^-2, … ,8^-15

Informacje techniczne:

- Kod pisany w C++

Wyniki dla typu float

LP.	x	f(x)	g(x)	|f(x) – g(x)|
1.	8^{-1}	0.00778222	0.00778222	2.32831e-09
2.	8^{-2}	0.00012207	0.000122063	7.45058e-09
3.	8^{-3}	1.90735e-06	1.90735e-06	1.81899e-12
4.	8^{-4}	0	2.98023e-08	2.98023e-08
5.	8^{-5}	0	4.65661e-10	4.65661e-10
6.	8^{-6}	0	7.27596e-12	7.27596e-12
7.	8^{-7}	0	1.13687e-13	1.13687e-13
8.	8^{-8}	0	1.77636e-15	1.77636e-15
…	…	…	…	…
23.	8^{-23}	0	1.43493e-42	1.43493e-42
24.	8^{-24}	0	2.24208e-44	2.24208e-44
25.	8^{-25}	0	0	0

Wnioski dla float

W czwartym kroku możemy zauważyć, że wyniki obydwóch funkcji różnią się od siebie znacząco. Wynika to z tego że dla typ float w języku c++ nie jest typem dokładnym, ma dokładność do 8 cyfr znaczących, możemy to zauważyć po różnicach jakie występują między obydwoma funkcjami do momentu wyzerowania f, oraz zajmuje 32 bity. Podczas dodawania nasz argument zanika, różni się znacząco od 1 i podczas tego dodawania traktowany jest jak 0, przez to że argumenty sprowadzane są do tej samej cechy i nasze wartości są gubione.

Wnioski dla float

Ten sam problem jest z funkcją g, natomiast nie jest on tak bardzo widoczny dla początkowych argumentów. Wpływa on tak naprawdę tylko na dzielnik, gdzie znowu pod pierwiastkiem w dodawaniu argument zanika, z czego powstaje 1. Można powiedzieć że nasza funkcja g tymczasowo będzie postaci g(x) = x^2/2. Dla odpowiednio małych argumentów, potęgowanie również doprowadza do wyzerowania argumentu.

Wyniki dla typu double

LP.	x	f(x)	g(x)	|f(x) – g(x)|
1.	8^{-1}	0.00778222	0.00778222	6.50521e-17
2.	8^{-2}	0.00012207	0.000122063	8.32803e-17
3.	8^{-3}	1.90735e-06	1.90735e-06	3.46945e-18
4.	8^{-4}	2.98023e-08	2.98023e-08	1.32349e-23
5.	8^{-5}	4.65661e-10	4.65661e-10	1.0842e-19
6.	8^{-6}	7.27596e-12	7.27596e-12	2.64698e-23
7.	8^{-7}	1.13687e-13	1.13687e-13	6.46235e-27
8.	8^{-8}	1.77636e-15	1.77636e-15	1.57772e-30
9.	8^{-9}	0	2.77556e-17	2.77556e-17
…	…	…	…	…
177.	8^{-177}	0	1.01185e-320	1.01185e-320
178.	8^{-178}	0	1.58101e-322	1.58101e-322
179.	8^{-179}	0	0	0
180.	8^{-180}	0	0	0
181.	8^{-181}	0	0	0

Wnioski dla double

Typ double jest typem dokładniejszym od float, możemy to stwierdzić po różnicach w wartościach naszych funkcji, oraz po tym że zanik argumentu w sumowaniu pod pierwiastkiem następuje dopiero w 9 kroku, czyli dla argumentu 8^(-9). Typ ten zajmuje 64 bity, ma dokładność do 16 cyfr znaczących. Dla funkcji g wyniki róże od 0 obserwujemy do kroku 178, potem następuje wyzerowanie argumentu w dzielnej, ponieważ wpadamy w dziurę pomiędzy 0 a pierwszą dobrze reprezentowaną liczbą w zakresie, następuje zaokrąglenie z niedomiarem.

Wyniki dla typu long double

LP.	x	f(x)	g(x)	|f(x) – g(x)|
1.	8^{-1}	0.00778222	0.00778222	5.2516e-20
2.	8^{-2}	0.00012207	0.000122063	2.37169e-20
3.	8^{-3}	1.90735e-06	1.90735e-06	8.27181e-24
4.	8^{-4}	2.98023e-08	2.98023e-08	1.32349e-23
5.	8^{-5}	4.65661e-10	4.65661e-10	2.52435e-29
6.	8^{-6}	7.27596e-12	7.27596e-12	2.64698e-23
7.	8^{-7}	1.13687e-13	1.13687e-13	6.46235e-27
8.	8^{-8}	1.77636e-15	1.77636e-15	1.57772e-30
9.	8^{-9}	2.77556e-17	2.77556e-17	3.85186e-34
10.	8^{-10}	4.33681e-19	4.33681e-19	9.40395e-38
11.	8^{-11}	0	6.77626e-21	6.77626e-21
…	…	…	…	…
357.	8^{-357}	0	7.81123e-646	7.81123e-646
358.	8^{-358}	0	1.2205e-647	1.2205e-647
359.	8^{-359}	0	0	0

Wnioski dla long double

Long double jest najdokładniejszym typem zmiennoprzecinkowym w języku c++, co widać po wynikach pomiarów, x^2 pod pierwiastkiem zanika dopiero w kroku 11, dodawanie dla tego typu jest działaniem w miarę dokładnym dla argumentów rzędu około 10^-20. Typ ten zajmuje 80 bitów. Dla funkcji g wyniki nie zerowe dla argumentów rzędu 8^-358.

Wyniki dla dużych argumentów (najciekawsze)

Typ argumentu	Wartość argumentu	f(x)	g(x)	|f(x) – g(x)|
Float	8^4	4095	4095	0.000244141
Float	8^21	9.22337e+18	9.22337e+18	0
Float	8^22	inf	nan	nan
Double	8	7.06226	7.06226	8.88178e-16
Double	8^2	63.0078	63.0078	7.10543e-15
Double	8^170	3.35195e+153	3.35195e+153	0
Double	8^171	inf	nan	nan
Long Double	8	7.06226	7.06226	8.67362e-19
Long Double	8^2	63.0078	63.0078	3.46945e-18
Long Double	8^341	8.98847e+307	8.98847e+307	0
Long Double	8^342	inf	nan	nan

Obliczanie wyniku dla dużych argumentów

Dla dużych argumentów nastąpi problem wyjścia z zakresu, nasz argument dla komputera będzie tożsamy z nieskończonością, przez co wartości funkcji f będą tożsame z nieskończonością. Natomiast wartości dla funkcji g będą utożsamiane jako nan (nie liczba), ponieważ mamy do czynienie z dzieleniem dwóch wartości które dla danej reprezentacji są już nieskończonością.

Jak obliczać z kolei wartości dla dużych argumentów?

Jeżeli chcemy policzyć wartość dla dużego argumentu np. bliskiego największej wartości double, musimy wybrać typ zmiennej z większym zakresem, w przypadku tego języka long double.