Obliczenia Naukowe
lista 1

Mateusz Tofil

19 października 2021

1 Zadanie 1

1.1 Opis zadania

W tym zadaniu należało wyznaczyć *macheps* czyli najmniejszą liczbę macheps, która spełnia następująca nierówność $fl(1.0+macheps) > 1.0$. Następnie trzeba było wyznaczyć liczę *eta*, czyli najmniejszą liczbę większą od 0 tj. $eta > 0.0$. Kolejnym zadaniem, było wyznaczenie największej liczby *max*. Każdą z tych liczb (czyli. *macheps, eta, max*) należało wyznaczyć itercyjnie dla wszystkich typów zmiennopozycyjnych.

1.2 Metoda rozwiązania

W pliku `zad1.jl` znajdują się programy, z których otrzymałem wyniki przeprowadzonych badań. W każdym z podproblemie zasada działania była bardzo podobna i polegała na dzieleniu lub mnożeniu liczby początkowej, aż do momentu kiedy nie zostało spełnione zdanie logiczne.

Badałem liczbę *macheps* w pętli, aż do momentu kiedy nie zaszedł warunek $1 + current/2 \neq 1$. W momencie spełnienia warunku, pętla kończyła swoją pracę i zwraca aktualna wartość dla której warunek zachodzi, czyli *macheps*.

Analogiczny algorytmy wykorzystałem do wyznaczenia liczby *eta*. W każdej iteracji dzieliłem liczbę do momentu kiedy nie zaszedł warunek $current/2 \neq 0$.

Licząc liczbę maksymalną zatrzymujemy się po osiągnięciu nieskończoności, która w rzeczywistości została przekroczona. Po wyjściu z pętli należy dodawać do liczby połowy przerwy między nieskończonością. Operacje powtarzać, aż do momentu gdy $\frac{x}{2^k} \geq 1$ dla pewnego k.

1.3 Otrzymane wyniki

W tabelach o numerach 1, 2, 3 zaprezenotwałem zestawienia wyników, które otrzymałem z przeprowadzonych przeze mnie badań. Porównuje je z budowanymi funkcjami w języku Julia, takimi jak np. *eps*() czy $nextfloat()$) Otrzymane wyniki są zgodnę z wbudowanymi funkcjami, co jednoznacznie stwierdza, że napisane przeze mnie funkcje są poprawnie napisane i zwracają poprawne wyniki.

typ	moja funkcja	funkcja *eps*()	float.h
Float16	0.000977	0.000977	b.d.
Float32	1.1920929e-7	1.1920929e-7	1.1920928955e-07
Float64	2.220446049250313e-16	2.220446049250313e-16	2.2204460493e-16

Tablica 1: Wartości epsilona maszynowego dla typów zmiennopozycyjnych

typ	moja funkcja	funkcja $nextfloat(0.0)$
Float16	6.0e-8	6.0e-8
Float32	1.0e-45	1.0e-45
Float64	5.0e-324	5.0e-324

Tablica 2: Wartości *eta* dla typów zmiennopozycyjnych

typ	moja funkcja	funkcja $nextfloat(0.0)$
Float16	6.55e4	6.55e4
Float32	3.4028235e38	3.4028235e38
Float64	1.7976931348623157e308	1.7976931348623157e308

Tablica 3: Wartości max dla typów zmiennopozycyjnych

1.3.1 Macheps a precyzja arytmetyki

Z wykładu wiemy, że precyzja arytmetyki to $\frac{1}{2}\beta^{t-1}$. Liczba bitów przeznaczona na zapisanie cześci ułamkowej to właśnie $t-1$. Dla typu pojedycznej prezycji przeznaczone jest 24-bity, natomiast dla podwójnej już 53-bity. Podstawiając, dane to wzoru wyżej, otrzymujemy:
DOKOŃCZYĆ———————————

1.3.2 *eta* a liczba MIN_{sub}

Liczba *eta* i liczba MIN_{sub} leżą w tym samym rzędzie, a różnica między nimi jest bardzo mała.

1.3.3 Związek między `floatmin()`, a liczbą MIN_{nor}

Podobnie jak liczby *eta* i MIN_{sub}, wartości zwracane przez funkcje `floatmin()` leżą w tym samym rzędzie co liczba MIN_{nor}

typ	funkcja $floatmin()$	MIN_{nor}
Float32	1.1754944e-38	1.2e-38
Float64	2.2250738585072014e-308	2.2e-308

Tablica 4: Porównanie wartości floatmin() i MIN_{nor}

1.4 Wnioski

2 Zadanie 2

2.1 Opis zadania

Zadanie to polegało na sprawdzeniu, czy jesteśmy w stanie obliczając wartość wyrażenia a) $3 * (\frac{4}{3} - 1) - 1$ otrzymać wartość epsilona maszynowego.

2.2 Metoda rozwiązania

Napisałem 3 funkcję, dla każdego typu `Float16, Float32, Float64` funckję, która obliczała wyżej wymienione wyrażenie.

2.3 Otrzymane wyniki

Otrzymane wyniki porównałem z wcześniejszymi wynikami z poprzednich zadań i zestawiłem w tabeli 5.

typ	wynik wyrażenia	*eps*
Float16	-0.000977	0.000977
Float32	1.1920929e-7	1.1920929e-7
Float64	-2.220446049250313e-16	2.220446049250313e-16

Tablica 5: Porównanie wartości z wyrażenia a) z *eps*

Jak widać część naszych wyników pokrywają się. W miejsach gdzie wyniki nie zgadzają się, można zauważyć, są to liczby przeciwne. Najprawdopodobniej spowodowane jest to tym że liczba $\frac{4}{3}$ w rozwinięciu binarnym ma nieskończone rozwinięcie. Rozwinięcie to prezentuje się następująco 1.(10). Liczba bitów znaczących dla typów danych wynosi:

- `Float16` - 10 bitów
- `Float32` - 23 bity
- `Float64` - 52 bity

Więc dla typu Float16 i Float64 ostatnią cyfrą mantysy jest 0, w przeciwieństwie do typu Float32, gdzie ostatnia cyfra mantysy to 1. To właśnie decyduje o znaku odejmowania.

2.4 Wnioski

Żyjąc w świecie gdzie istnieje tylko skończona dokładność reprezentacji, niektóre równania dające w matematyce tożsamości, w arytmetyce zmiennoprzecinkowej mogą dawać zupełnie różne wyniki.

3 Zadanie 3

3.1 Opis problemu

W tym zadaniu, problem z jakim musiałem się zmierzyć to było zbadanie rozmieszczenia liczb zmiennoprzecinkowych w arytmetyce IEEE 754 o podwójnej prezycji. Rozmieszczenie liczb należało przebadać na różnych przedziałach liczbowych.

3.2 Rozwiązanie