Sprawozdanie
Obliczenia naukowe - lista 1

Witold Karaś

zadanie 1

Program oblicza epsilon maszynowy (*macheps*), liczbę maszynową *eta* oraz największą liczbę jaką można zapisać jako float. Wykonuje te operacje odpowiednio dla typów Float16, Float32 oraz Float64.

Rozwiązanie
a) epsilon maszynowy jest obliczany iteracyjnie. W pętli sprawdzana jest prawdziwość warunku $1.0 + \epsilon > 1.0$ i co iteracje ϵ jest zmiejszany o połowę. Precyzja arytmetyki informuje nas o odległościach między kolejnymi liczbami w arytmetyce a epsilon maszynowy określa odległość między 1 i następną liczbą x taką że x ¿ 1

typ zmiennej	*macheps*	nextfloat	`float.h`
Float16	0.000977	0.000977	nie występuje
Float32	1.1920929e-7	1.1920929e-7	1.192092896e-07F
Float64	2.220446049250313e-16	2.220446049250313e-16	2.2204460492503131e-016

Tabela 1: wartości z wyjścia programu `zadanie1.jl`

b) eta obliczana jest iteracyjnie. W pętli dzielona jest przez 2 jeśli prawdziwy jest warunek $\eta/2 > 0$.
Eta określa doległość między zerem a następną liczbą x taką że $x > 0$ czyli jest MIN_{SUB}. Wbudowana funkcja **floatmin** według specyfikacji zwraca najmniejszą liczbę dodatnią w postaci zmiennopozycyjnej, jednak jest ona większa od uzyskanych wyników. Wynika to z faktu że zwracana wartość przez **floatmin** jest w postaci normalnej.

typ zmiennej	*obliczona eta*	floatmin	eta
Float16	6.0e-8	6.104e-5	6.0e-8
Float32	1.0e-45	1.1754944e-38	1.0e-45
Float64	5.0e-324	2.2250738585072014e-308	5.0e-324

Tabela 2: wartości z wyjścia programu `zadanie1.jl`

c) wartość maksymalną dla typu zmiennopozycyjnego jest obliczana iteracyjnie poprzez mnożenie przez 2. rozpoczynamy od wartości prevfloat(one(type))ponieważ jest złożona z samych jedynek w części ułamkowej. Wykładnik podczas mnożenia przez 2 zmienia się jedynie wykładnik dzięki czemu stosunkowo szybko program znajduje wartość maksymalną.

typ zmiennej	obliczona wartość	floatmax	`float.h`
Float16	6.55e4	6.55e4	nie występuje
Float32	3.4028235e38	3.4028235e38	3.402823466e+38F
Float64	1.7976931348623157e308	1.7976931348623157e308	1.7976931348623158e+308

Tabela 3: wartości z wyjścia programu `zadanie1.jl`

Poniższy fragment przedstawia definicje znalezione w pliku `float.h`

```
//dla typu float
#define FLT_EPSILON      1.192092896e-07F        // smallest
    such that 1.0+FLT_EPSILON != 1.0
#define FLT_MAX          3.402823466e+38F        // max
    value
#define FLT_MIN          1.175494351e-38F        // min
    normalized positive value
#define FLT_TRUE_MIN     1.401298464e-45F        // min
    positive value
//dla typu double
#define DBL_EPSILON      2.2204460492503131e-016 // smallest
    such that 1.0+DBL_EPSILON != 1.0
#define DBL_MAX          1.7976931348623158e+308 // max
    value
#define DBL_MIN          2.2250738585072014e-308 // min
    positive value
#define DBL_TRUE_MIN     4.9406564584124654e-324 // min
    positive value
```

Wnioski Znalezione przez program liczby są zgodne z wynikami funkcji wbudowanych.

zadanie 2

Program oblicza *macheps* metodą Kahana. Otrzymane wyniki są, z dokładnością do znaku, zgodne z tymi które otrzymałem w zadaniu 1

zadanie 3

Program sprawdza czy eksponenty skrajnych liczb z przedziału są identyczne, jeżeli są oznacza to że liczby w przedziale są równomiernie rozmieszczone.

typ zmiennej	*output*
Float16	-0.000977
Float32	1.1920929e-7
Float64	-2.220446049250313e-16

Tabela 4: wartości z wyjścia programu `zadanie2.jl`

Następnie ze wzoru $\delta = 2^{eksponent-1023} * 2^{-52}$ jest obliczana δ która reprezentuje odstęp między liczbami w zakresie.
Wnioski:
Liczby w artytmetyce zmiennopozycyjnej nie są rozmieszczone równomiernie na osi liczbowej. Im są bliżej 0 tym są bliżej siebie, jednak nie osiągają 0 (ograniczeniem dolnym jest liczba eta z zadania 1)

zakres	*output*
(0.5, 1.0)	1.1102230246251565e-16
(1.0, 2.0)	2.220446049250313e-16
(2.0, 4.0)	4.440892098500626e-16

Tabela 5: wartości z wyjścia programu `zadanie3.jl`

zadanie 4

Sprawdziłem metodą brute force - iteracyjnie po $x \in (1, 2)$ jaka jest najmniejsza liczba spełniająca nierówność $x \cdot \frac{1}{x} \neq 1$
output: 1.000000057228997

zadanie 5

Przetestowałem obliczanie wektora skalarnego 4 wskazanymi metodami. Przez fakt, że iloczyny były liczbami o bardzo różnym rzędzie wartości (bardzo małe i bardzo duże liczby) a precyzja arytmetyki jest ograniczona.

zadanie 6

Zaprogramowałem dwie funkcje

$$f(x) = \sqrt{x^2 + 1} - 1$$

i

$$g(x) = \frac{x^2}{\sqrt{x^2 + 1} + 1}$$

typ zmiennej	*output*
Float32	-0.4999443
	-0.4543457
	-0.5
	-0.5
Float64	1.0251881368296672e-10
	-1.5643308870494366e-10
	0.0
	0.0

Tabela 6: wartości z wyjścia programu `zadanie5.jl`

obliczające to samo z punktu widzenia matematyki.
Ponieważ w funkcji $f(x)$, x dąży do 0 to wartość pierwiastka dąży do 1. W rezultacie odejmujemy od siebie bardzo bliskie liczby i przez stosowane obcięcie wynik szybko traci na dokładności.
W funkcji $g(x)$, błąd odejmowania już nie występuje dzięki czemu mamy zachowaną wyższą precyzję obliczeń, mimo że wyrażenie jest równoważne.

zadanie 7

Program oblicza pochodną funkcji w przybliżeniu $\widetilde{f}'(x) = \frac{f(x+h)-f(x)}{h}$ w punkcie $x = 1$ oraz dokładną wartość pochodnej $f'(x) = cos(x) - 3sin(3x)$
Dla dużych n, $h \to 0$, czyli w liczniku dostajemy bardzo zbliżone wartości i przez obcięcie odejmowania dostajemy 0, z tego samego powodu od pewnego momentu błąd ma prawie stałą wartość, ok 0.59.

n	$x = 8^{-n}$	$f(x)$	$g(x)$
0	1.0	0.41421356237309515	0.4142135623730951
1	0.125	0.0077822185373186414	0.0077822185373187065
2	0.015625	0.00012206286282867573	0.00012206286282875901
3	0.001953125	1.9073468138230965e-6	1.907346813826566e-6
4	0.000244140625	2.9802321943606103e-8	2.9802321943606116e-8
5	3.0517578125e-5	4.656612873077393e-10	4.6566128719931904e-10
6	3.814697265625e-6	7.275957614183426e-12	7.275957614156956e-12
7	4.76837158203125e-7	1.1368683772161603e-13	1.1368683772160957e-13
8	5.960464477539063e-8	1.7763568394002505e-15	1.7763568394002489e-15
9	7.450580596923828e-9	0.0	2.7755575615628914e-17
10	9.313225746154785e-10	0.0	4.336808689942018e-19
11	1.1641532182693481e-10	0.0	6.776263578034403e-21
12	1.4551915228366852e-11	0.0	1.0587911840678754e-22
13	1.8189894035458565e-12	0.0	1.6543612251060553e-24
14	2.2737367544323206e-13	0.0	2.5849394142282115e-26
15	2.842170943040401e-14	0.0	4.0389678347315804e-28
16	3.552713678800501e-15	0.0	6.310887241768095e-30
17	4.440892098500626e-16	0.0	9.860761315262648e-32
18	5.551115123125783e-17	0.0	1.5407439555097887e-33
19	6.938893903907228e-18	0.0	2.407412430484045e-35
20	8.673617379884035e-19	0.0	3.76158192263132e-37

Tabela 7: wartości z wyjścia programu `zadanie5.jl` (ograniczone do 21 pierwszych wyników)

n	$\widetilde{f}'(1.0), h = 2^{-n}$	$f'(1.0)$	$\lvert\widetilde{f}'(1.0) - f'(1.0)\rvert$
0	0.8218544151266947	0.11694228168853815	0.7049121334381565
1	0.7771241507177988	0.11694228168853815	0.6601818690292607
2	0.746269987940468	0.11694228168853815	0.6293277062519298
3	0.7278860798075346	0.11694228168853815	0.6109437981189965
4	0.7178180618879821	0.11694228168853815	0.600875780199444
5	0.7125457423057711	0.11694228168853815	0.5956034606172329
6	0.7098474639715988	0.11694228168853815	0.5929051822830607
7	0.7084824701063042	0.11694228168853815	0.591540188417766
8	0.7077959684277175	0.11694228168853815	0.5908536867391794
9	0.7074517112415606	0.11694228168853815	0.5905094295530224
10	0.7072793304150764	0.11694228168853815	0.5903370487265382
11	0.7071930768624952	0.11694228168853815	0.5902507951739571
12	0.7071499342910101	0.11694228168853815	0.590207652602472
13	0.7071283590557869	0.11694228168853815	0.5901860773672487
14	0.7071175704477355	0.11694228168853815	0.5901752887591973
15	0.7071121758999652	0.11694228168853815	0.5901698942114271
16	0.7071094785642345	0.11694228168853815	0.5901671968756963
17	0.7071081298636273	0.11694228168853815	0.5901658481750891
18	0.7071074555278756	0.11694228168853815	0.5901651738393374
19	0.7071071183308959	0.11694228168853815	0.5901648366423577
20	0.7071069497615099	0.11694228168853815	0.5901646680729717
21	0.7071068654768169	0.11694228168853815	0.5901645837882787
22	0.7071068231016397	0.11694228168853815	0.5901645414131016
23	0.707106800749898	0.11694228168853815	0.5901645190613598
24	0.7071067914366722	0.11694228168853815	0.5901645097481341
25	0.7071067839860916	0.11694228168853815	0.5901645022975535
26	0.7071067690849304	0.11694228168853815	0.5901644873963923
27	0.7071067690849304	0.11694228168853815	0.5901644873963923
28	0.7071067690849304	0.11694228168853815	0.5901644873963923
29	0.7071067094802856	0.11694228168853815	0.5901644277917475
30	0.7071065902709961	0.11694228168853815	0.5901643085824579
31	0.7071065902709961	0.11694228168853815	0.5901643085824579
32	0.7071065902709961	0.11694228168853815	0.5901643085824579
33	0.7071056365966797	0.11694228168853815	0.5901633549081415
34	0.7071037292480469	0.11694228168853815	0.5901614475595087
35	0.7070999145507812	0.11694228168853815	0.5901576328622431
36	0.7071075439453125	0.11694228168853815	0.5901652622567743
37	0.70709228515625	0.11694228168853815	0.5901500034677118
38	0.70709228515625	0.11694228168853815	0.5901500034677118
39	0.70703125	0.11694228168853815	0.5900889683114618

40	0.70703125	0.11694228168853815	0.5900889683114618
41	0.70703125	0.11694228168853815	0.5900889683114618
42	0.70703125	0.11694228168853815	0.5900889683114618
43	0.70703125	0.11694228168853815	0.5900889683114618
44	0.70703125	0.11694228168853815	0.5900889683114618
45	0.703125	0.11694228168853815	0.5861827183114618
46	0.703125	0.11694228168853815	0.5861827183114618
47	0.6875	0.11694228168853815	0.5705577183114618
48	0.6875	0.11694228168853815	0.5705577183114618
49	0.625	0.11694228168853815	0.5080577183114618
50	0.5	0.11694228168853815	0.38305771831146185
51	0.5	0.11694228168853815	0.38305771831146185
52	0.0	0.11694228168853815	0.11694228168853815
53	0.0	0.11694228168853815	0.11694228168853815
54	0.0	0.11694228168853815	0.11694228168853815

Tabela 8: wartości z wyjścia programu `zadanie5.jl`