Porównajmy teraz	że stworzona funkcja znanazła identyc z te wartości z epsilonami z float.h $N=1.1920928955078125e-7$	zzny epsilon jak funkcja wbudowana w język julia
DBL_EPSILO	N=1.1920928955078125e-70 $N=2.2204460492503131e-16$ 6czba macheps z precyzją arytmetyki?	3^{1-t}
Float16: $2^{-1} * 2^{1}$ Float32: $2^{-1} * 2^{1}$ Float64: $2^{-1} * 2^{1}$ In [69]:	$e^{-23} = 2^{-23}$ $e^{-52} = 2^{-52}$	liczba cyfr mantysy
println("Float32 println("Float64 Float16: 0.00097 Float32: 1.19209 Float64: 2.22044	29e-7	st równy jej precyzji
Stostujemy tutaj In [46]: function eta(typeta = type(1)		•
<pre>end return eta end for t in [Float] println("ary println("fur print("next end arytmetyka:Float</pre>	<pre>continuous properties of the continuous properties of the continuous</pre>	
func eta:6.0e-8 next float:6.0e-6 arytmetyka:Float func eta:1.0e-45 next float:1.0e-6 arytmetyka:Float func eta:5.0e-32 next float:5.0e-3	32 45 64 4	
Zacznijmy od teg $nast$ ępujący $MIN_{sub}=2^{1-t}$ c_{min} to minimaln c	$st~2^{c_{min}}$	można przedstawić w postaci nieznormalizowanej, wzór jest $-2^{d-1}+2$, d to liczba bitów przeznaczonych na ceche $-10*2^{-14}=2^{-24}$
Float64: $c_{min} = -\frac{1}{2}$ In [71]: println("Float16 println("Float64	$-2^{8-1}+2=-126=>MIN_{sub}=2$ $-2^{11-1}+2=-1022=>MIN_{sub}=2$ 6: ", Float16(2^-24)) 2: ", Float32(2^-149)) 4: ", Float64(2^-1074))	
	sperymentu możemy wyczytać, że M	IN_{sub} jest równy liczbie η ch arytmetyk, oraz jaki ma związek z liczbą MIN_{nor}
println("Float32 println("Float64 Float16: 6.104e- Float32: 1.17549 Float64: 2.22507	44e-38 38585072014e-308 est to najmniejsza liczba, którą można	przedstawić w postaci znormalizowanej, wyrażana wzorem
println("Float32	$\lambda = 2^{-126}$	
Float16: 6.104e-Float32: 1.17549 Float64: 2.22507 Zauważyć można	5 44e-38 38585072014e-308 , że wartości zwracane przez funkcje fl	
Stosujemy podob do zapisania w da In [68]: function find_ma max = prevfl while isinf(max = ty	ne iteracyjne podejście jak w poprzedr anej arytmetyce	ą da sie zapisać w podanej arytmetyce nich podpunktach, czyli zwiększamy liczbe dopóki jest możliwa
<pre>println("ary println("fur print("float end arytmetyka:Float func find_max:6.</pre>		
floatmax:6.55e4 arytmetyka:Float func find_max:3. floatmax:3.40282 arytmetyka:Float func find_max:1. floatmax:1.79769	32 4028235e38 35e38 64 7976931348623157e308 31348623157e308	
$FLT_MAX = $ $DBL_MAX = $	1.7976931348623157 <i>e</i> 308	danie 2
$3*(rac{4}{3}-1)-$ In [30]: function kahan(t return type(end	na sprawdzeniu działania wzoru Kaha - 1 sype) 3.0)*(type(4.0)/type(3.0)-type(1.0	n'a.
<pre>for t in [Float] println(rpac println(rpac</pre>	0.000977 Float32	
kahan func epsil julia eps: arytmetyka: kahan func epsil julia eps:	on: 1.1920929e-7 1.1920929e-7 Float64 on: -2.220446049250313e-16 2.220446049250313e-16	na epsilonu maszynowywego otrzymanego ze wzoru Kahan'a
In [60]:	Zac distribution(start, step, en) .oat = start	danie 3
<pre>for i in 1:5 real_nex step_nex if(step_</pre>	<pre>ct_float = nextfloat(real_next_float ct_float += step next_float != real_next_float) irn false coat = en coat = en</pre>	t)
step_nex if (step_	<pre>id it_float = prevfloat(real_next_float it_float -= step inext_float != real_next_float) irn false</pre>	t)
<pre>println("krok:",</pre>	.al:",p[1],",",p[2]) step) no rozmieszczone:",check_distribut 250313e-16	ion(Float64 (p[1]),step, Float64 (p[2])),"\n")
In [79]: for i in 1:10 println(bits end 00111111111110000	yglądają kolejne liczby w arytmetyce F string(1.0+eps(Float64)*i)) 0000000000000000000000000000000000	000000000000000000000000000000000000000
0011111111110000 00111111111110000 00111111	000000000000000000000000000000000000	
	00000000000000000000000000000000000000	0000000000101 0000000000110 0000000000
Analizując powyż $\delta=2^{cecha-bias}*$ $bias=2^{d-1}-1$	2^{1-t} eksperyment, w którym wyznaczamy te	0000000000101 0000000000110 0000000000
Analizując powyż $\delta=2^{cecha-bias}*$ $\delta=2^{cecha-bias}*$ $bias=2^{d-1}-1$ Przeprowadzmy e podanych przedzi In [99]: for p in [[0.5,1] # parsujemy # w arytmety exp = parse(bias = 2^10 step = 2.0^(println("prz	cooooooooooooooooooooooooooooooooooooo	00000000000000000000000000000000000000
Analizując powyż s $\delta=2^{cecha-bias}*$ $bias=2^{d-1}-1$ Przeprowadzmy e podanych przedziania podanych przedziani	00000000000000000000000000000000000000	00000000000000000000000000000000000000
Analizując powyżs $\delta = 2^{cecha-bias} *$ $bias = 2^{d-1} - 1$ Przeprowadzmy e podanych przedzi In [99]: for p in [[0.5,1] # parsujemy # w arytmety exp = parse(bias = 2^10 step = 2.0^(println("przprintln("kroprintln("czyprintln("c	coccooccoccoccoccoccoccoccoccoccoccocco	20000000000000000000000000000000000000
Analizując powyż s $\delta=2^{cecha-bias}*$ $bias=2^{d-1}-1$ Przeprowadzmy e podanych przedzialin [99]: for p in [[0.5,1] # parsujemy # w arytmety exp = parse(bias = 2^10 step = 2.0^(println("prz println("kroprintln("czy) end przedzial:0.5,1.4 krok:1.110223024 czy równo rozmie przedzial:2.0,4.4 krok:4.440892098 czy równo rozmie przedzial:2.0,4.1 krok:4.440892098 czy równo rozmie pr	poodoooooooooooooooooooooooooooooooooo	2000000000000101 200000000000111 200000000
Analizując powyż s $\delta=2^{cecha-bias}*$ $bias=2^{d-1}-1$ Przeprowadzmy e podanych przedzia In [99]: for p in [[0.5,1] # parsujemy # w arytmety exp = parse(bias = 2^10 step = 2.0^(println("prz println("kro println("czy) end przedzial:0.5,1.4 krok:1.110223024 czy równo rozmie przedzial:2.0,4.4 krok:4.440892098 czy równo rozmie Drzedzial:2.0,4.4 krok:4.440892098 czy równo rozmie Zadanie polegało $x*(\frac{1}{x}) \neq 1$ Algorytm polega jest tożsamość x więc naturalnie n najmniejszą taką In [25]: let liczba = Flo while prevfl liczba = end println("="^	2000000000000000000000000000000000000	popogogogogogogogogogogogogogogogogogog
Analizując powyż x $\delta = 2^{cecha-bias} *$ $bias = 2^{d-1} - 1$ Przeprowadzmy e podanych przedzialin [99]: for p in [[0.5,1] # parsujemy # w arytmety exp = parse(bias = 2^10 step = 2.0^(println("przeprintln("kroprintln("czy end przedzial:0.5,1 krok:1.110223024 czy równo rozmie przedzial:2.0,4 krok:4.440892098 czy równo rozmie przedzial:2.0,	poonoooooooooooooooooooooooooooooooooo	popogogogogogogogogogogogogogogogogogog
Analizując powyż s $\delta=2^{cecha-bias}*$ $bias=2^{d-1}-1$ Przeprowadzmy e podanych przedzialine podanych przedzialine podanych przedzialine podanych przedzialine podanych przedzialine println ("czy end przedzialine podanych podanych podanych podanych przedzialine podanych przedzialine podanych przedzialine podanych przedzialine podanych przedzialine podanych przedzialine podanych przedzialize podanych podanych przedzialize podanych podanych przedzialize podanych podanych przedzialize podanych przedzia	poooooooooooooooooooooooooooooooooooo	pegeogeoegolio pegeogeoegolii pegeogeoegolii pegeogeoegoliii pegeogeogeogeoliii pegeogeogeogeoliii pegeogeogeogeogeogeogeogeogeogeogeogeogeo
Analizując powyż: $\delta=2^{cecha-bias}*$ $bias=2^{d-1}-1$ Przeprowadzmy e podanych przedzialio podanych przedzialio podanych przedzialio println ("przeprintln ("przedzialio println ("czy end przedzialio println ("czy end przedzialio prz	peoeoeoeoeoeoeoeoeoeoeoeoeoeoeoeoeoeoeo	popogogogogogogogogogogogogogogogogogog
Analizując powyż: $\delta = 2^{cecha-bias} *$ $bias = 2^{d-1} - 1$ Przeprowadzmy e podanych przedziz In [99]: for p in [[0.5,1] # parsujemy # w arytmety exp = parsety bias = 2^10 step = 2.0^(println("czy) end Przedzial:0.5,1.' end Przedzial:0.5,1.' rock:1.110223024 czy równo rozmie przedzial:2.0,4.' krok:1.110223024 czy równo rozmie przedzial:2.0,4.' krok:4.440892098 czy równo rozmie Zadanie polegało $x * (\frac{1}{x}) \neq 1$ Algorytm polega jest tożsamość x więc naturalnie n najmniejszą taką In [25]: let liczba = Flowinie println("=""" end println("=""" end Zadanie polegało sposoby $\sum_{i=1}^{n} x_i * y_i$ Najpierw mnożym while prevfluczba = end println("=""" end Zadanie polegało sposoby $\sum_{i=1}^{n} x_i * y_i$ Najpierw mnożym wartości dodatnic obie sumy częście Ostatni sposób je wartości sumujen end function sum_bac sum type (contact obie sumy częście) Ostatni sposób je wartości sumujen end function sum_bac sum type (contact obie sumy częście) Ostatni sposób je wartości sumujen end function sum_bac sum type (contact obie sumy częście) Ostatni sposób je wartości sumujen end function sum_bac sum type (contact obie sumy częście) Ostatni sposób je wartości sumujen end function sum_bac sum type (contact obie sumy częście) Ostatni sposób je wartości sumujen end function sum_bac sum type (contact obie sumy częście)	poecessoeseoeseoeseoeseoeseoeseoeseoeseoe	poepoepoepoeloeloeloeloeloeloeloeloeloeloeloeloelo
Analizując powyż: $\delta = 2^{cecha-bias} *$ $bias = 2^{d-1} - 1$ Przeprowadzmy e podanych przedzi In [99]: for pin [[0.5,1] # parsujemy # w arytmety exp = parse bias = 2^10 step = 2.0^(println("prz println("krc println("czy) end przedzial:0.5,1.4 krok:1.110223024 czy równo rozmie przedzial:2.0,4.6 krok:1.110223024 czy równo rozmie przedzial:2.0,4.6 krok:4.440892098 czy równo rozmie Drzedzial:2.0,4.7 krok:4.440892098 czy równo rozmie Drzedzial:2.0,4.7 krok:4.440892098 czy równo rozmie Zadanie polegało println("="^println("="println("=	peopogeoeoooooooooooooooooooooooooooooo	poeceeseesessis poeceeses processis
Analizując powyż: $\delta = 2^{cecha-bias} *$ $bias = 2^{d-1} - 1$ Przeprowadzmy e podanych przedzi In [99]: for p in [[0.5,1] # parsujemy # w arytmety exp = parse(bias = 2.0^(println("prz println("prz println("czy end przedzial:0.5,1.4 krok:1.110223024 czy równo rozmie Zadanie polegało $x * (\frac{1}{x}) \neq 1$ Algorytm polega jest tożsamość x więc naturalnie n najmniejszą taką In [25]: let liczba = Flo while nextfl liczba = Flo while nextfl liczba = end println("="" println("="" end Zadanie polegało soposoby $\sum_{i=1}^{n} x_i * y_i$ Najpierw mnożym wartości odotarico obie sumy częścio Ostatni sposób je wartości sumujen Zadanie polegało sposoby $\sum_{i=1}^{n} x_i * y_i$ Najpierw mnożym wartości odotarico obie sumy częścio Ostatni sposób je wartości sumujen In [19]: function sum_for sum = type(6 for i in 1:1 sum += x end return sum end function sum_des xy = x .* y xy_pos = sor xy_pos sum = for i in xy xy_pos = sor xy_neg = sor	peogeoeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeee	### Special Company of the Company o
Analizując powyż: $\delta = 2^{cecha-bias} *$ $bias = 2^{d-1} - 1$ Przeprowadzmy e podanych przedzi podanych przedzi podanych przedzi podanych przedzia podanych podanych przedzia podanych	peopeoeoeoeoeoeoeoeoeoeoeoeoeoeoeoeoeoe	Signature 4 Initiation (Float64(p[1]), step, Float64(p[2])), "\n") Janie 4 Initiation (Float64(p[1]), step, Float64(p[2])), "\n") Janie 5 Janie 7 Janie 8 Janie 9 Janie 10 Janie 9 Janie 9 Janie 10 Janie 9 Janie 10 Jan
Analizujac powyż: $\delta = 2^{cecha-bias} *$ $bias = 2^{d-1} - 1$ Przeprowadzmy ce podanych przedzia in [99]: for p in [[0.5,1] # parsujemy # w arytmety exp = parse (bias = 2^0) cept in ["przeprintln ("przeprintln ("przeprintln ("czy) end przedzial: 0.5, 1. krok: 1.11023024 czy równo rozmie corzedzial: 0.4. krok: 4.44089298 czy równo rozmie corzedzial: 0.4. krok: 4.440892998 czy równo rozmie corzedzial: 0.4. krok: 4.44089299 czy równo rozmie corzedzial: 0.4. krok: 4.44089299 czy równo rozmie corzedzial: 0.4. krok: 4.44089299 czy równo rozmie corzedzial: 0.4. krok: 4.4408929 czy równo rozmie c	poeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeee	### Seesenance 10
Analizując powyż. \$ = 2^{cecha-bias}** bias = 2^{d-1} - 1 Przeprowadzmy e podanych przedzia In [99]: for p in [[0.5, 1." # parsujemy # warytmety exp = parse(bias = 2.10 step = 2.00 printla("prz printla("rz printla("rz printla("rz printla("sz end przedzial:0.5, 1." krok:1.110223024 czy równo rozmie przedzial:2.0, 4." krok:4.440892098 czy równo rozmie przedzial:2.0, 4." krok:4.40892098 czy równo rozmie printl("pad" printl("pad" println("="" end Function sum_for sum + yi end return sum end function sum_for sum + yi end return sum end function sum_des yy pos_sum = for i in ky xy pos = sor xy_neg = sor x	### ### ### ### ### ### ### ### ### ##	### Seeseeseeseeseeseeseeseeseeseeseeseesees
Analizując powyż $\delta = 2^{cecha-bias} *$ $bias = 2^{d-1} - 1$ Przeprowadzmy e podanych przedzi podanych prze	peegeoegoegoegoegoegoegoegoegoegoegoegoeg	### Seeseeseeseeseeseeseeseeseeseeseeseesees
Analizując powyż: $\delta = 2^{cecha-bias} *$ $bias = 2^{d-1} - 1$ Przeprowadzmy e podanych przedzia: for pin [[0.5,1] # parsujemy # warytmety exp parsed bias = 2.0 (println("parprintln("czy end przedzial:0.5,1 krok:1.11023024czy równo rozmie przedzial:2.0,4 krok:4.44882098czy równo rozmie przedzial:2.0,4 krok:4.4982098czy równo rozmie przedzial:2.0,4 krok:4.44882098czy równo rozmie przedzial:2.0,4 krok:4.44882098czy równo rozmie przedzial:2.0,4 krok:4.44882098czy równo rozmie println("parprintln("p	pageogeogeogeogeogeogeogeogeogeogeogeogeog	processor of the company of the comp
Analizując powyż: $\delta = 2^{cecha-bias} *$ $bias = 2^{d-1} - 1$ Przeprowadzmy e podanych przedzi In [99]: for pin [[0.5,1] # parsujemy # w ytmety exp = parse(bias = 2.0^c) println("prz println("prz println("prz println("prz println("cz) end przedzia::0.5,1.1 krok:110223024c cyr okuno rozmie cyr okuno rozmie Zadanie polegało x* (\frac{1}{x}) \neq 1 Algorytm polega jest tożsamość x więc naturalnie n najmniejszą taką In [25]: let liczba = Flo white prevfl	pageogoogoogoogoogoogoogoogoogoogoogoogoog	processor state of the control of th
Analizujac powyż: $\delta = 2^{\text{cecha}-bias}*$ $bias = 2^{d-1} - 1$ Przeprowadzmy e podanych przedziach podanych przedzianiach powyż w warywney exp println("kreprint	poseosoosoosoosoosoosoosoosoosoosoosoosoo	processor of the company of the comp
Analizując powyże Analizując powyże $\delta = 2^{\text{dech}a - \text{bias}} *$ $bias = 2^{\text{def}} - 1$ Przeprowadzmy podanych przedzia In [99]: for p in [[0.5,1] # parsujemy # warytmey exp parse(bias = 2.10 cyrintin [90]: for pin [[0.5,1] # parsujemy # warytmey exp parse(bias = 2.10 cyrintin [90]: for pin [[0.5,1] # parsujemy # warytmey exp z.00 cyrintin [90]: for pin [[0.5,1] # parsujemy # warytmey exp z.00 cyrintin [90]: for pin [[0.5,1] krok: 4.1023024 krok: 4.1023024 krok: 4.40802098 czy równo rozmie Zadanie polegało $x * (\frac{1}{x}) \neq 1$ Algorytm polega jest tożsamość x więc naturalnie n najmniejszą taką In [25]: let liczba = Fle white nextft liczba = Fle	poseosoosoosoosoosoosoosoosoosoosoosoosoo	inconsociation in a consideration of the constraint of the consociation in a consociation of the consociat
Analizując powyżz $\delta = 2^{\operatorname{cecha}-\operatorname{bias}} *$ $\delta = 2^{\operatorname{da}-1} - 1$ Przeprowadznyce prodanych przedz In [99]: for pin [[0.5,1] ## ar surjem ## ar s	December	in earthod sumujomy w kignotci rosagogi, natornate ujomne
Analizując powyże $\delta = 2^{\text{cecha}-\text{bias}} *$ $\delta = 2^{\text{cecha}-\text{bias}} *$ $\delta = 3^{\text{cecha}-\text{bias}} $	poseeseoseoseoseoseoseoseoseoseoseoseoseo	services (1) is a serviced by service transmisser with the service of the service
Analizując powyż 2 $\delta = 2^{cecha-bias}*$ $bias = 2^{d-1} - 1$ Przeprowadzny podanych przedzi podanych prz	poepoepoepoepoepoepoepoepoepoepoepoepoep	selections (Compared to the selection of
Analizując powyże $\delta = 2^{\text{cecha}}$ $\delta = 3^{\text{cecha}}$ $\delta = 3^{ce$	poseenoseoseoseoseoseoseoseoseoseoseoseoseoseo	and is provided by state of section reason when we have a provided provided by state of section reason when we have a provided by the state of section reason when we have a provided by the section of section reason when a provided water is provided by (1, 2), Netro spelling secondary or provided by (1, 2), Netro spelling secondary or provided by the section of section reason which is provided by the section of section by the section of section in section of section reason which is provided by the section of section reason is provided by the section of section is provided by the section of sectio
Analizując powyże A = 2 eccha-bias * bias = 2 d-1	possessonoonoonoonoonoonoonoonoonoonoonoonoon	second complete or another programme to the following per of certy in the following per of certy in control per of cert
Analizując powyż $\delta = 2^{\text{decha-bias}} *$ $b^{\text{ials}} = 2^{\text{d}-1} - 1$ Przeprowadzmy e podanych przedzi podanych przedzi podanych przedzi podanych przedzi podanych przedzi podanych przedzi podanych przedzial podanych println ("przeprintln ("przeprin	### ### ### ### ### ### ### ### ### ##	consequence (as a control of the con
Analizujac powyż \$ = 2ccha - was * bias = 2ccha - was * bias = 2d-1 - 1 Przepanych przedz In [99]: [19]: [10]: [10]: [10]: [10]: [11]: [12]: [12]: [12]: [12]: [13]: [14]: [14]: [15]: [16]: [17]: [18]: [1	### ### ### ### ### ### ### ### ### ##	Section 1.
Analizujac powyż Analizujac powyż \$ = 2°ceha - bas * bias = 2°d - d - 1 Programy - d - 1 Programy - d - 1 Programy - d - 1 In [99]: In [99	poepocoepocoepocoepocoepocoepocoepocoep	Innie 6 a core dans according worth of the property of the core according to the core a
Analizuja powyż Analizuja powyż \$ = 2 cecha - bias * bias = 2 d - 1	pospoooooooooooooooooooooooooooooooooo	Jamie 4 India 4 India 4 India 4 India 4 India 5 India 6 India 6 India 6 India 7 India 6 India 7 India 6 India 7 India 6 India 7 India 8 India 7 Ind
Analizując powyż \$ = 2cecha - bia * bias = 2d - 1 Przeprowadzny cy # arytmety # arytmet	pocococococococococococococococococococ	In the Company of the

Mateusz Nasewicz

Zadanie 1

W pierwszym podpunkcie mamy wyznaczyć epsilon maszynowy, czyli odległość między jedynką a następną liczbą w

Korzystamy ze wskazówki i szukamy iteracyjnie najmniejszej liczby takiej że $1+\epsilon>1$

podanej arytmetyce

zmniejszanie h nie	1.5 1.125 1.125 1.0625 1.03125 1.015625 1.0078125 1.0078125 1.00839625 1.00939625 1.00939625 1.00939625 1.00939625 1.00909765625 1.009044140625 1.0090249140625 1.0090276763125 1.00906103515665 1.00908152587890655 1.00909152587890655 1.00909152587890655 1.00909152587890655 1.00909152587890655 1.00909152587890655 1.00909152587890655 1.00909152587890655 1.00909152587890656 1.0090909174663371582 1.0090909174663371582 1.0090909174663371582 1.0090909174663371582 1.0090909174663371582 1.0090909174663371582 1.00909091746911612 1.009090909746566 1.009090909746566 1.00909090909156448 1.00909090974595866 1.00909090974595866 1.009090909115615 1.00909090909115615 1.00909090909115615 1.0090909090915615 1.0090909090915615 1.00909090909016515 1.00909090909016515 1.00909090909016145 1.009090909090915 1.009090909090915 1.0090909090909015 1.0090909090909016 1.0090909090909091 1.0090909090909091 1.0090909090909091 1.0090909090909091 1.0090909090909091 1.0090909090909091 1.0090909090909091 1.0090909090909091 1.0090909090909091 1.0090909090909091 1.0090909090909091 1.0090909090909091 1.0090909090909091 1.0090909090909091 1.0090909090909091 1.0090909090909091 1.009090909090909091 1.0090909090909091 1.0090909090909091 1.0090909090909091 1.0090909090909091 1.0090909090909091 1.0090909090909091 1.0090909090909091 1.0090909090909091 1.009090909090909091 1.009090909090909091 1.009090909090909091 1.00909090909090909090909090909090909090

h

h+1 _____