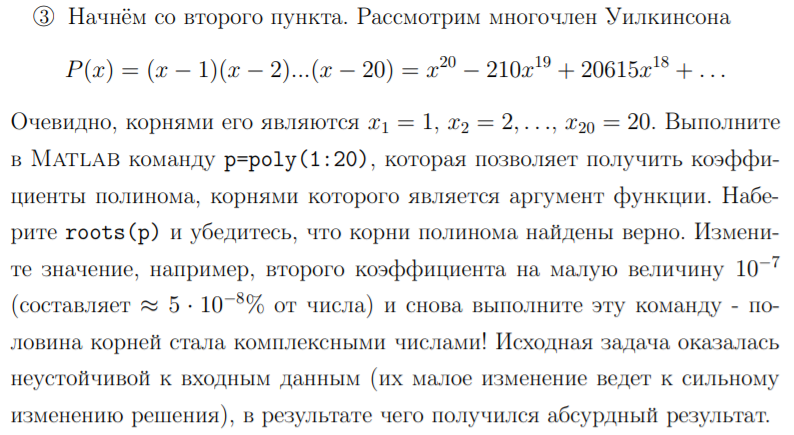
|  |
| --- |
| Лабораторная работа 1 |
| Распространение ошибок в вычислительных процедурах |
| Артамоновой Анастасии ПИН-24 |

|  |
| --- |
|  |



p = poly(1:20);

roots(p)

*ans =*

*20.0003*

*18.9972*

*18.0112*

*16.9711*

*16.0483*

*14.9354*

*14.0653*

*12.9491*

*12.0334*

*10.9840*

*10.0061*

*8.9984*

*8.0003*

*7.0000*

*6.0000*

*5.0000*

*4.0000*

*3.0000*

*2.0000*

*1.0000*

p(2) = p(2) + 10^(-7);10^(-17)

roots(p)

*ans =*

*20.4220 + 0.9992i*

*20.4220 - 0.9992i*

*18.1573 + 2.4702i*

*18.1573 - 2.4702i*

*15.3150 + 2.6988i*

*15.3150 - 2.6988i*

*12.8466 + 2.0627i*

*12.8466 - 2.0627i*

*10.9213 + 1.1037i*

*10.9213 - 1.1037i*

*9.5678*

*9.1137*

*7.9941*

*7.0002*

*6.0000*

*5.0000*

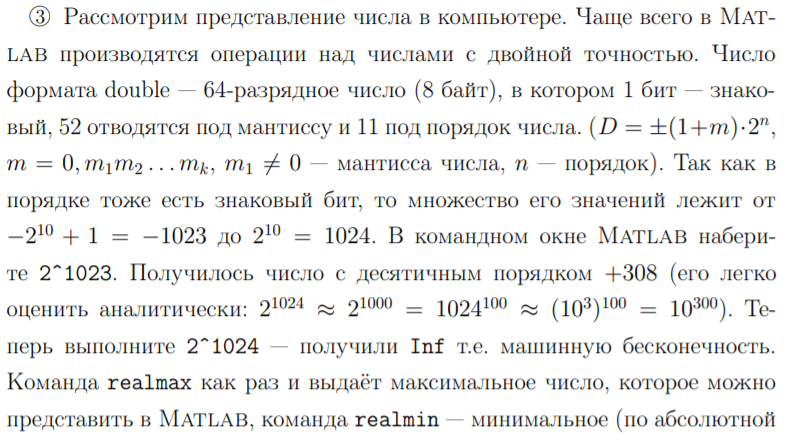
*4.0000*

*3.0000*

*2.0000*

*1.0000*

Решение неустойчиво, так как при малом изменении значения аргумента, значение функции сильно меняется.



2^1023

*ans = 8.9885e+307*

2^2024

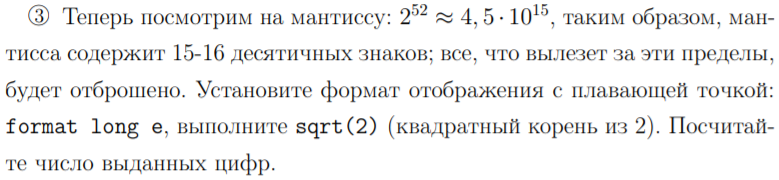
*ans = Inf*

realmax

*ans = 1.7977e+308*

realmin

*ans = 2.2251e-308*

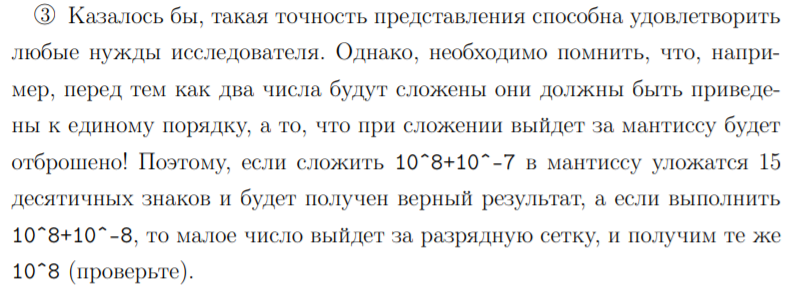
**

format long e

sqrt(2)

ans = 1.414213562373095e+000

Число выданных цифр равно 16.

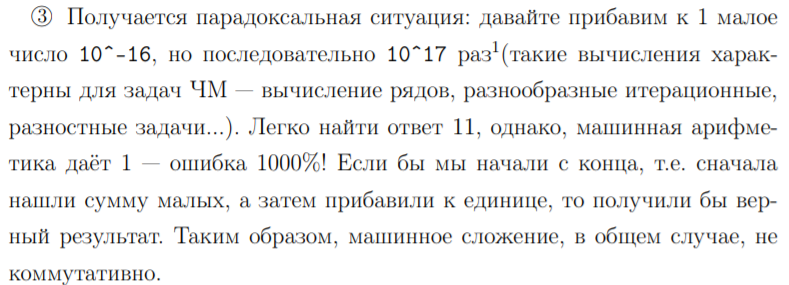


10^8 + 10^(-7)

*ans = 1.000000000000001e+008*

10^8 + 10^(-8)

*ans = 1.000000000000000e+008*

**

result = 1;

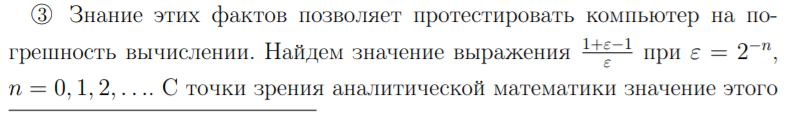
for i= 1:10^6

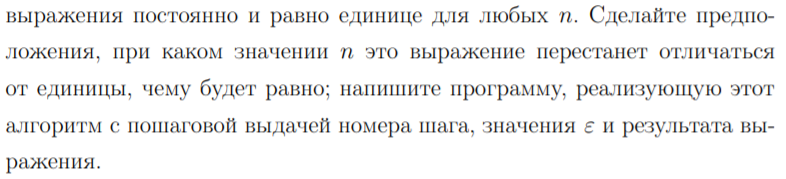
result = result + 10^(-5);

end

result

*result = 1.099999999975465e+001*

**

**

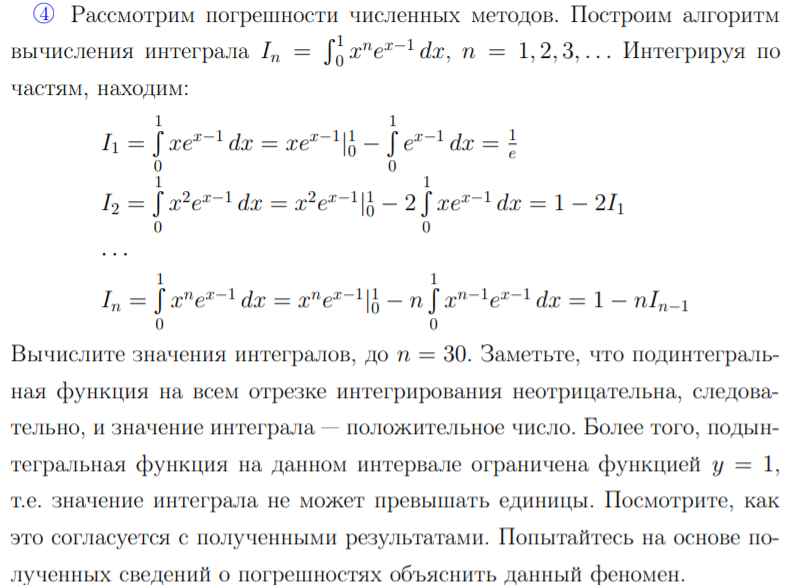
for n = 1:53 При n= 47 выражение равно 0

eps = 2^(-n);

(100+eps-100)/eps

end

При n = 53 выражение станет равно 0



syms x

I = 1/exp(1);

for i=2:30

sprintf('n = %d',i)

I = 1 - i\*I

end

n = 2

I = 2.642411176571154e-001

n = 3

I = 2.072766470286537e-001

n = 4

I = 1.708934118853853e-001

n = 5

I = 1.455329405730734e-001

n = 6

I = 1.268023565615595e-001

n = 7

I = 1.123835040690837e-001

n = 8

I = 1.009319674473304e-001

n = 9

I = 9.161229297402684e-002

n = 10

I = 8.387707025973157e-002

n = 11

I = 7.735222714295276e-002

n = 12

I = 7.177327428456692e-002

n = 13

I = 6.694743430062999e-002

n = 14

I = 6.273591979118010e-002

n = 15

I = 5.896120313229858e-002

n = 16

I = 5.662074988322274e-002

n = 17

I = 3.744725198521337e-002

n = 18

I = 3.259494642661593e-001

n = 19

I = -5.193039821057027e+000

n = 20

I = 1.048607964211406e+002

n = 21

I = -2.201076724843952e+003

n = 22

I = 4.842468794656693e+004

n = 23

I = -1.113766822771040e+006

n = 24

I = 2.673040474650495e+007

n = 25

I = -6.682601176626236e+008

n = 26

I = 1.737476306022821e+010

n = 27

I = -4.691186026251618e+011

n = 28

I = 1.313532087350553e+013

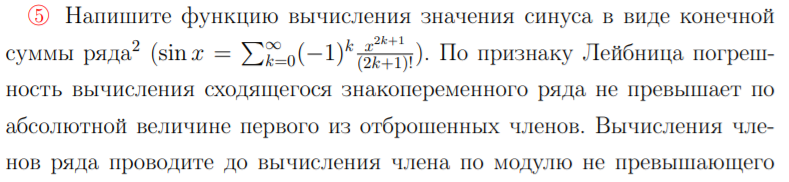
n = 29

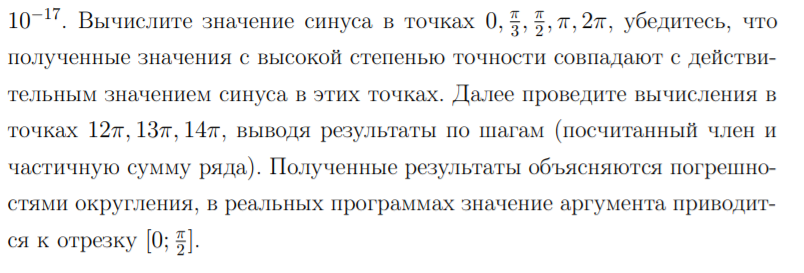
I = -3.809243053316594e+014

n = 30

I = 1.142772915994978e+016

1/e не является конечной дробью и MATLAB округляет это значение до 16 знаков. Погрешность при вычислении I1 на каждом следующем этапе умножалась на число, модуль которого больше единицы, что приводит к такому результату. Следовательно данное решение неустойчиво.





Функция:

result = 0;

i = 0;

while(true)

f = (-1)^i \* x^(2\*i + 1) / factorial(2\*i + 1);

result = result + f;

if (abs(f) < 10^(-17))

break;

end

i = i + 1;

end

result

sin(x)

При x = 0

result = 0

Проверка

ans = 0

При x = pi/3

result = 8.660254037844385e-001

Проверка

ans = 8.660254037844386e-001

При x = pi/2

result = 1.000000000000000e+000

Проверка

ans = 1

При x = pi

result = 3.328027611925569e-016

Проверка

ans = 1.224646799147353e-016

При x = 2pi

result = 3.300897561490086e-015

Проверка

ans = -2.449293598294706e-016

**При х = 12pi**

Шаг 0

f =

3.769911184307752e+001

result =

3.769911184307752e+001

Шаг 1

f =

-8.929807683926347e+003

result =

-8.892108572083269e+003

Шаг 2

f =

6.345624183707595e+005

result =

6.256703097986762e+005

Шаг 3

f =

-2.147273155583299e+007

result =

-2.084706124603431e+007

Шаг 4

f =

4.238547317337191e+008

result =

4.030076704876848e+008

Шаг 5

f =

-5.476291888243632e+009

result =

-5.073284217755947e+009

Шаг 6

f =

4.989123186633167e+010

result =

4.481794764857571e+010

Шаг 7

f =

-3.376503233854057e+011

result =

-2.928323757368300e+011

Шаг 8

f =

1.764251532907330e+012

result =

1.471419157170500e+012

Шаг 9

f =

-7.331564081604559e+012

result =

-5.860144924434059e+012

Шаг 10

f =

2.480901844343075e+013

result =

1.894887351899669e+013

Шаг 11

f =

-6.968211157451131e+013

result =

-5.073323805551462e+013

Шаг 12

f =

1.650563700175192e+014

result =

1.143231319620045e+014

Шаг 13

f =

-3.341622719903058e+014

result =

-2.198391400283012e+014

Шаг 14

f =

5.848757610408251e+014

result =

3.650366210125239e+014

Шаг 15

f =

-8.938052725562344e+014

result =

-5.287686515437105e+014

Шаг 16

f =

1.202932425236984e+015

result =

6.741637736932738e+014

Шаг 17

f =

-1.436668294789759e+015

result =

-7.625045210964853e+014

Шаг 18

f =

1.532902456774330e+015

result =

7.703979356778443e+014

Шаг 19

f =

-1.470037975755849e+015

result =

-6.996400400780045e+014

Шаг 20

f =

1.273934043683861e+015

result =

5.742940036058565e+014

Шаг 21

f =

-1.002516282597193e+015

result =

-4.282222789913366e+014

Шаг 22

f =

7.195955719916363e+014

result =

2.913732930002996e+014

Шаг 23

f =

-4.730369111489184e+014

result =

-1.816636181486188e+014

…

Шаг 65

f = -3.727452375943002e-016

result = 1.140295178893491e-001

Шаг 66

f = 3.017510351971951e-017

result = 1.140295178893492e-001

Шаг 67

f = -2.370677289564583e-018

result = 1.140295178893492e-001

Проверка

ans = -1.469576158976824e-015

**При х = 13pi**

Шаг 0

f =

4.084070449666731e+001

result =

4.084070449666731e+001

Шаг 1

f =

-1.135346497776979e+004

…

result =

-1.131262427327312e+004

Шаг 71

f =

-6.315063064558125e-018

result =

1.318057553492365e+001

Проверка

ans = -1.960672839908942e-015

**При х = 14pi**

Шаг 0

f = 4.398229715025710e+001

result = 4.398229715025710e+001

Шаг 1

f = -1.418020386845712e+004

result =

-1.413622157130686e+004

…

Шаг 75

f = -1.581589619913186e-017

result = 4.451056211091088e+002

Шаг 76

f = 1.315571946675626e-018

result = 4.451056211091088e+002

Проверка

ans = -1.714505518806294e-015

При вычислении 2pi, нам понадобится меньше шагов, чем для вычисления 14pi. При этом при больших аргументах слагаемые ряда имеют существенно различный порядок. Это может вызвать потерю значащих цифр при округлении значения мантиссы. Из-за этого может получиться неправильный ответ. Поэтому для тригонометрических функций следует использовать формулы приведения.

У нас получились такие результаты из-за погрешностей округления и из-за того, что в реальных программах значение аргумента приводиться к отрезку [0,pi/2].