Nam Nguyen

npn190000

Exercises

Section 1.1: 1, 5(b,c,e), 8 (b), 12(c,d)

1. In high school, some students have been misled to believe that 22/7 is either the actual value of π or an acceptable approximation to π. Show that 355/113 is a better approximation in terms of both absolute and relative errors. Find some other simple rational fractions n/m that approximate π. For example, ones for which |π −n/m| < 10−9. Hint: See Problem 1.1.4.

**Answer**:

. pi space equals space 3.1415927

. 355/113 = 3.1415929

. 22/7 = 3.1428571

epsilon subscript 1 equals open vertical bar pi space minus fraction numerator space 22 over denominator 7 end fraction close vertical bar space almost equal to 1.2 space times space 10 to the power of negative 3 end exponent
epsilon subscript 2 equals open vertical bar pi minus 355 over 113 close vertical bar space almost equal to space 2.7 space times space 10 to the power of negative 7 end exponent




355/113 a better approximation of pi

eta subscript 1 equals open vertical bar fraction numerator pi space minus fraction numerator space 22 over denominator 7 end fraction over denominator pi end fraction close vertical bar space almost equal to 4.0 space times space 10 to the power of negative 4 end exponent
eta subscript 2 equals open vertical bar fraction numerator pi minus 355 over 113 over denominator pi end fraction close vertical bar space almost equal to space 8.5 space times space 10 to the power of negative 8 end exponent




355/113 has lower relative error in appropriate pi

Example:

 we can use n, x is any number like (9n pi) / 9n = 3.141592654. Then |π −n/m| < 10−9

5. A given doubly subscripted array (ai j )n×n can be added in any order. Write the pseudocode segments for each of the following parts. Which is best?

Answer:

b. sum from j space equals space 1 to n of sum from i space equals space 1 to n of space a subscript i j end subscript

**integer** sum, j, I, n

sum = 0

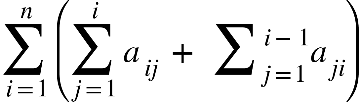
**for** j = 1 **to** n **do**

**for** i = 1 **to** n **do**

sum = sum + aji

**end for**

**end for**

c. 

**integer** sum, j, I, n

sum = 0

**for** i = 1 **to** n **do**

**for** j = 1 **to** n **do**

sum = sum + aij

**end for**

**for** j = 1 **to** i-1 **do**

sum = sum + aji

**end for**

**end for**

e. sum from k space equals space 2 to 2 n of sum from i space plus space j space equals space k to n of a subscript i j end subscript

**integer** sum, x, y, k, j, I, n

sum = 0

**for** i = 1 **to** n **do**

j = n + 1 – i

sum = sum + aij

**end for**

**for** k = 2 **to** n **do**

**for i** = 1 **to** n-1 **do**

j = k-i

x = n+1 -i

y = n+1 -j

sum = sum + aij + axy

**end for**

**end for**

In each example it has there own algorithm. By the consider time worst, then sample b is les time than another one.

8. Show how these polynomials can be efficiently evaluated:

b. p(x) = 3(x − 1)5 + 7(x − 1)9

**Answer:**

b. p(x) = 3(x − 1)5 + 7(x − 1)9

= 3(x − 1)5 + 7(x − 1)4(x - 1)5

= (x-1)5 + (3 + 7(x − 1)4)

Let z = (x-1)

Then p(x) = z5 + (3+7z4)

We can compute z5 = z4 \* z using one multiplication.

Then we compute 3 + 7\*z4 , and then multiply the result by z5.

12. Using summation and product notation, write mathematical expressions for the following

pseudocode segments:

**c. integer** *i, n*; **real** *v, x*; **real array** *(ai )0:n*

*v* ← a*0*

**for** *i* = 1 **to** *n* **do**

*v* ← *vx* + *ai*

**end for**

**Answer:**

v0 = a0.

v1 = a0 x + a1.

v2 = (a0 x + a1)x + a2 = a0 x2 + a1x + a2 .

…

Vn = a0 xn + a1xn-1 + … +an .

V = sum from i equals 0 to n of a subscript n minus i end subscript x to the power of i

**d. integer** *i, n*; **real** *v, x, z*; **real array** *(ai )*0:*n*

*v* ← *a0*

*z* ← *x*

**for** *i* = 1 **to** *n* **do**

*v* ← *v* + *zai*

*z* ← *xz*

**end for**

**Answer:**

v0 = a0.

v1 = a0 + a1x

v2  = a2 x2 + a1x + a0 .

…

Vn = an xn + an-1xn-1 + … +a0 .

V = sum from i equals 0 to n of a subscript i x to the power of i

**Section 1.2: 1, 4(d,f), 14, 44**

1. The Maclaurin series for (1 + x)n is also known as the binomial series. It states that

left parenthesis 1 space plus space x right parenthesis to the power of n space equals space 1 space plus space n x space plus space fraction numerator n left parenthesis n space minus space 1 right parenthesis over denominator 2 factorial end fraction space space x 2 space plus fraction numerator space n left parenthesis n space minus space 1 right parenthesis left parenthesis n space minus space 2 right parenthesis over denominator 3 factorial end fraction space x 3 space plus times space times space times space left parenthesis x squared space less than space 1 right parenthesis

Derive this series. Then give its particular forms in summation notation by letting

n = 2, n = 3, and n = 1/2 . Next use the last form to compute √1.0001 correct to 15 decimal places (rounded).

**Answer:**

Let f(x) = (1+x)n

Then f’(x) = n(1+x)n-1 .

Then f’’(x)= n(n-1)(1+x)n – 2 .

Then f’’’(x) = n(n-1)(n-2)(1+x)n – 3

Then f(4)(x) = n(n-1)(n-2)(n-3)(1+x)n – 4

We have Maclaurin:

f(x) = f(0) + f’(0)x + f’’(0)x2/2! + f’’’(x)x3/3! + …

So: f(0) = 1; f’(0) = n; f’’(0) = n(n-1); f’’’(0)= n(n-1)(n-2); f((4)(x)= n(n-1)(n-2)(n-3)

Therefore,

f(x) = 1 + nx + n(n-1)x2/2! + n(n-1)(n-2)x3/3! + …

Hence, (1+x)n = 1 + nx + n(n-1)x2/2! + n(n-1)(n-2)x3/3! + … (x2 < 1)

For n = 2: then (1+x)2 = 1 + 2x + x2

For n = 3: then (1+x)3 = 1 + 3x + 3x2+ x3

For n = 1/2: then (1+x)1/2 = 1 + (1/2)x - (1/23)x2 + (1/24)x3 + …

Then with n = ½, choose x = 0.0001

(1+.0001)1/2 = 1 + (1/2)0.0001 - (1/23)0.00012 + (1/24)0.00013 + … = 1.0000499987500625

Round off value we have (1.0001)1/2 = **1.000049998750062**

4. Why do the following functions not possess Taylor series expansions at x = 0?

**Answer:**

d. f (x) = cot x

-> f(x) = f(0) + f’(0)x + f’’(0)x2/2! + f’’’(x)x3/3! + …

We have:

f(x) = cot x

f’(x) = -csc2 x then f’(0) = -∞

Hence, Taylor Series about x = 0 does not exit as the first derivative f’(0) = -∞

f. f (x) = xπ

-> f(x) = f(0) + f’(0)x + f’’(0)x2/2! + f’’’(x)x3/3! + …

We have:

f(x) = xπ

f’(x) = π (x)3.14 -1

f’’(x) = π (2.14) (x)3.14 -2

f’’’(x) = π (2.14)(1.14) (x)3.14 -3

f(4)(x) = π (2.14)(1.14)(0.14) (x)3.14 -4

When x = 0, at f(4)(0) = -∞

at x=0, the function become -∞

Hence, Taylor Series about x = 0 does not exit as the function become -∞ when x = 0

14. Write the Taylor series for the function f (x) = x3 −2x2 +4x −1, using x = 2 as the

point of expansion; that is, write a formula for f (2 + h).

**Answer:**

f(2+h) = f(2) + hf’(2) + (h2/2!)f’(2) + (h3/3!)f’’(2) + (h4/4!)f(4) (2)

f (x) = x3 −2x2 +4x −1 then f(2) = 7

f’ (x) = 3x2 −4x +4 then f’(2) = 8

f’’ (x) = 6x −4 then f’’(2)= 8

f’’’ (x) = 6 then f’’’(2) = 6

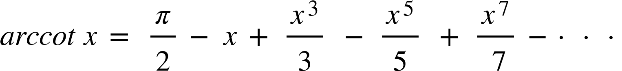
f(4) (x) = 0 then f(4) (2) = 0

f(2+h) = 7 + 8h + (h2/2!)8 + (h3/3!)6 + (h4/4!)0

= 7 + 8h + 4h2 + h3

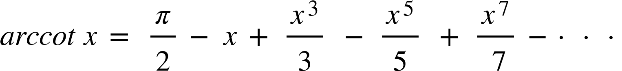
Hence, f(2+h) = 7 + 8h + 4h2 + h3

44. How many terms are needed in the series

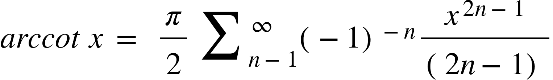
to compute arccot x for x2 < 1 accurate to 12 decimal places (rounded)?

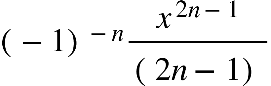
**Answer:**

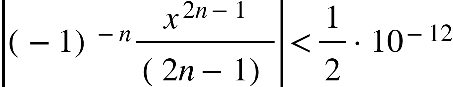
Consider:

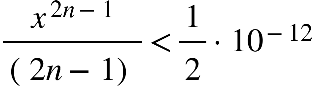


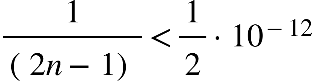
That is:

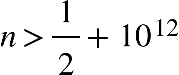


Therefore, the (n+1)th term 

We have for accurate to 12 decimal places : 

Now: 

For x2 < 1: 

Then: 

Hence, at least 1012 +1 terms are need in the series to compute arc cot x, for accurate to 12 decimal places.

**Section 2.1: 1(a,b), 3, 5(e-h), 12, 20, 36**

1. Determine the machine representation in single precision on a 32-bit word-length computer

for the following decimal numbers.

a. 2−30

b. 64.01562 5

Answer:

a. 2−30 = (-1)0 \* 2127 -30 \* (1.0)

97/2 = 1

48/2 = 0

24/2 = 0

12/2 = 0

6/2 = 0

3/2 =1

½ = 1

Read from bottom to top (1100001)2. We have substitute binary form 297 is

0 01100001 00000000000000000000000

Hence, the single precision machine representation in a 32 bit word length computer 2−30 is

0 01100001 00000000000000000000000

b. 64.015625

(64)10 = (1000000)2

64/2 = 0

32/2 = 0

16/2 = 0

8/2 = 0

4/2 = 0

2/2 = 0

½ = 1

(.015625)10 = (000001)2

.015625 \*2 = 0

.03125 \*2 = 0

.0625 \* 2 = 0

0.125\*2 = 0

0.25\*2 = 0

0.5\*2 = 1

(64.015625)10 = (1000000. 000001)2 = (1.000000 000001 \* 2 6)2 = (-1)0 \* 2127+6 \* 2-127 (1. 000000 000001)

= (-1)0 \* 2133 \* 2-127 (1. 000000 000001)

(133)­10 = (10000101)2

133/2 = 1

66/2 = 0

33/2 = 1

16/2 = 0

8/2 = 0

4/2 = 0

2/2 =0

½ = 1

We have substitute binary form 2133 is

0 10000101 000000 00000100000000000

Hence, the single precision machine representation in a 32 bit word length computer 64.015625 is

0 10000101 000000 00000100000000000

3. Which of these are machine numbers?

a. 10403

b. 1 + 2−32

c. 1/5

d. 1/10

e. 1/256

**Answer:**

a. 10403 Since 103 almost equal to210 , 10403 almost equal to21340. This is beyond range of the exponent even in double -precision. So 10403 is not a machine number

b. 1 + 2−32 has a normalized mantissa of 1.00000000000000000000000000000001 exactly. This require 32 bits of mantissa, which can be done in double – precision. So . 1 + 2−32 is a machine number.

c. 1/5 is not a finite of power of 2 because must have denominator that is power of 2 in reduce form. Hence 1/5 is not a finite of power of 2. We consider that 1/5 is not a machine learning.

d. 1/10 is not a finite of power of 2 because must have denominator that is power of 2 in reduce form. Hence 1/10 is not a finite of power of 2. We consider that 1/10 is not a machine learning.

e. 1/256 is 1\*2­-8 and can be represented exactly within 23 bit mantissa. This is a machine number

5. Identify the floating-point numbers corresponding to the following bit strings:

e. 0 00000001 00000000000000000000000

h. 0 01111011 10011001100110011001100

**Answer:**

e. 0 00000001 00000000000000000000000

Since 00000001 the actual biased exponent is 1-127 = -126. The sign is positive. So represented number 1\* 2-126.

h. 0 01111011 10011001100110011001100

Sign is positive

01111011 = 243 the actual biased exponent is 123-127 = -4.

1.10011001100110011001100

Since 1.100 re present for 3/5 = 1.5

The geometric series eevaluate to 3/2(1+1/16 +1/162 +1/162 +1/163 +1/164 + 1/165) = 1.599999905

Hence, we have 1.599999905 \* 2-4  = 0.099999995

12. What are the machine numbers immediately to the right and left of 2m? How far is each

from 2m?

**Answer:**

1 + epsilon ≠ 1

On a floating point number on a 32-bit word-length machine, epsilon = 2-23.

2m represented on the machine by an effective exponent of m, and mantissa of 1.0 (empty mantissa).

. The smallest machine number to its right given by machine by incrementing the rightmost bit in mantissa:

x = 2m  + (1 + epsilon).

x- 2m = 2m\*epsilon

On the a floating point number on 32 bit word length machine, this distance is 2m -23.

. The largest machine number to the left is given by decrementing the exponent, but using a full mantissa (1.111…..1):

y = 2m-1(2-epsilon) the distance from 2m

2m – y = 2m-1\*epsilon

On the a floating point number on 32 bit word length machine, this distance is 2m -24.

20. What is the roundoff error when we represent 2−1 +2−25 by a machine number? Note:

This refers to absolute error, not relative error.

**Answer**:

x = 2−1 +2−25 = 2−1 (1 +2−24) is not a machine number since the mantissa require at least 24 bits, whereas we only have 23 bits allocated in a single precision floating point number representation.

The two closet machine number are the one to its right x+ = 2−1 (1 +2−23)

and the one to its left. x- = 2−1 (1 + 0)

Turn out that they are equally distance from x, and the distance is

2-1(1+2-24) - 2−1 (1 + 0) = 2-25

36. Show by an example that in computer arithmetic a +(b+c) may differ from (a + b) + c.

**Answer:**

1 + epsilon ≠ 1

Let a = 1; b = epsilon; c = - epsilon (something can cause overfloat)

Consider a + (b + c) = 1

Consider (a +b) = 1 + epsilon (cause overfloat)

(a + b) + c = 1 – epsilon (error here already) ) ≠ 1

Hence, a + (b + c) = 1 while (a + b) + c ) ≠ 1 , then a + (b + c) ≠ (a + b) + c

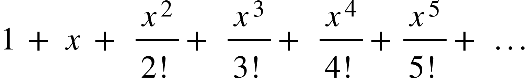
**Section 2.2: 2, 3, 9, 14, 17**

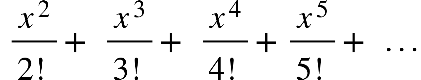
2. Calculate f (10−2) for the function f (x) = ex − x − 1. The answer should have five significant figures and can easily be obtained with pencil and paper. Contrast it with the straightforward evaluation of f (10−2) using e0.01 ≈ 1.0101.

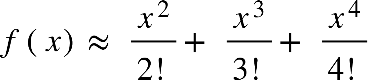
**Answer:**

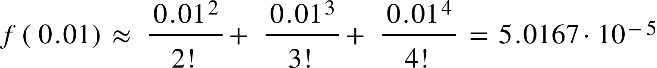
f (x) = ex − x – 1

Taylor series expansion for ex is

ex = 

then f(x) = ex − x – 1 = 

The answer should have five significant figures 



Instead, we use the approximation

e0.01 ≈ 1.0101.

f(0.01) ≈ 1.0101 – 0.01 -1 = 0.0001.

3. What is a good way to compute values of the function f (x) = ex − e if full machine precision is needed? Note: There is some difficulty when x = 1.

**Answer:**

When x = 1 then ex = e

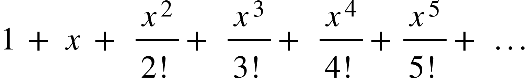
f (x) = ex – e then function cannot compute reliability.

When x> 1 then ex > e. We use the loss of precision theorem to determine the range of x in which at miss one bit lost in the subtraction f (x) = ex – e.

Text

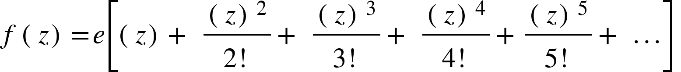
Description automatically generated

½ < 1 – (e/ex ) -> 1 + ln2 ≤ x .Hence, when x is not close to 1 in particular when x≤ 1-ln 2 or 1+ln 2 ≤x we can use original expression to compute f(x).

ex = 

f open parentheses x close parentheses equals e open square brackets open parentheses x minus 1 close parentheses plus space fraction numerator open parentheses x minus 1 close parentheses squared over denominator 2 factorial end fraction plus space fraction numerator open parentheses x minus 1 close parentheses cubed over denominator 3 factorial end fraction plus space fraction numerator open parentheses x minus 1 close parentheses to the power of 4 over denominator 4 factorial end fraction plus fraction numerator open parentheses x minus 1 close parentheses to the power of 5 over denominator 5 factorial end fraction plus space... close square brackets space

Let z = x-1



We can take for example 100 terms to get the full machine precision

There will be some difficulty when x=1 because the series may not be summable

9. For some values of x, the assignment statement y ← 1 − cos x involves a difficulty. What is it, what values of x are involved, and what remedy do you propose?

**Answer**:

When x ≈ 2πn -> cos(x) ≈ 1, so y = 1 – cos(x) cannot be computed reliability

When x> 1 then 1 – cos(x). We use the loss of precision theorem to determine the range of x in which at miss one bit lost in the subtraction f (x) = 1 – cos(x).

Loss of precision theorem: ½ ≤ 1 – cos(x) -> cos(x) ≤ ½ . then x outside the range [-π/3 + 2n π, π/3 + 2n π]

We have f (x) = (1 – cos(x))\*1.

= (1 – cos(x))\*( 1 + cos(x))/( 1 + cos(x))

= sin2 x /(1 + cos(x))

When x in range [-π/3 + 2n π, π/3 + 2n π] with f (x) = sin2 x /(1 + cos(x)). Otherwise, we can use the original expression as-is to compute f(x) = y.

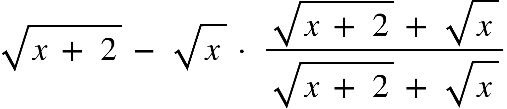
14. How can values of the function f(x)= square root of x space plus space 2 end root space minus space square root of x be computed accurately when x is large?

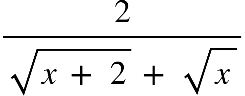
**Answer:**

when x is large square root of x space plus space 2 end root space almost equal to space square root of x. Since square root of x space plus space 2 end root space minus space square root of x cannot be computed reliability

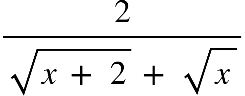
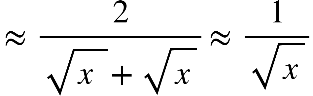
To avoid subtraction of two close numbers,

f(x) = square root of x space plus space 2 end root space minus space square root of x \* 1

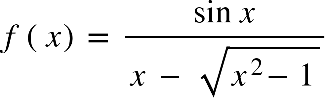
= 

= 

When x is large, we can further write

f(x) =   to avoid numerical loss

17. Without using series, how could the function

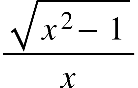
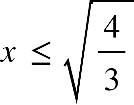


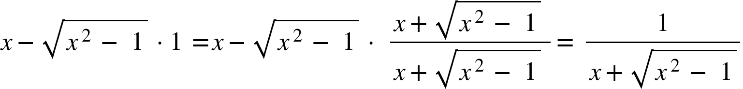
be computed to avoid loss of significance?

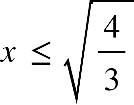
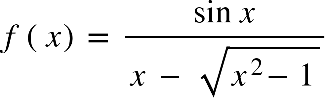
**Answer:**

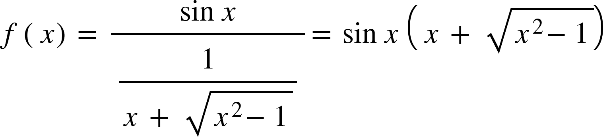
When x is large square root of x squared space minus 1 end root space almost equal to space x. Since x minus square root of x squared space minus space 1 end root space space cannot be computed reliability

Loss of precision theorem:

½ ≤ 1 -  ->  in the range in which th original expression will not suffer too much loss of precision.

We have: x minus square root of x squared space minus space 1 end root space space= 

When  we compute  but use instead

otherwise to avoid numerical loss.

Computing Exercises

Section 1.1: 1,2, 21(a,b)

1. Write and run a computer program that corresponds to the pseudocode program First

described in the text (p. 10) and interpret the results.

Answer:

Code (Mathlab):

n=30;

h=1;

emax=0;

x=0.5;

for i = 1:n

h=0.25 .\* h;

y=(sin(x + h) - sin(x))/h;

error = abs(cos(x) - y);

fprintf('%d %f %f %f\n',i,h,y,err);

if error > emax

emax=error;

imax=i;

end

end

disp('imax and emax values are');

fprintf('%d %f\n',imax,emax);

Output:

1 0.250000 0.808853 0.068730  
2 0.062500 0.862034 0.015548  
3 0.015625 0.873801 0.003781  
4 0.003906 0.876644 0.000939  
5 0.000977 0.877348 0.000234  
6 0.000244 0.877524 0.000059  
7 0.000061 0.877568 0.000015  
8 0.000015 0.877579 0.000004  
9 0.000004 0.877582 0.000001  
10 0.000001 0.877582 0.000000  
11 0.000000 0.877583 0.000000  
12 0.000000 0.877583 0.000000  
13 0.000000 0.877583 0.000000  
14 0.000000 0.877583 0.000000  
15 0.000000 0.877583 0.000000  
16 0.000000 0.877583 0.000000  
17 0.000000 0.877583 0.000000  
18 0.000000 0.877583 0.000000  
19 0.000000 0.877579 0.000004  
20 0.000000 0.877563 0.000019  
21 0.000000 0.877441 0.000141  
22 0.000000 0.877930 0.000347  
23 0.000000 0.878906 0.001324  
24 0.000000 0.875000 0.002583  
25 0.000000 0.875000 0.002583  
26 0.000000 0.750000 0.127583  
27 0.000000 0.000000 0.877583  
28 0.000000 0.000000 0.877583  
29 0.000000 0.000000 0.877583  
30 0.000000 0.000000 0.877583  
27 0.877583

2. (Continuation) Select a function f and a point x and carry out a computer experiment like the one given in the text. Interpret the results. Do not select too simple a function. For example, you might consider 1/x, log x, ex , tan x, cosh x, or x3 − 23x.

**Answer:**

Code Matlab:

n=30;

h=1;

emax=0;

x=0.5;

for i = 1:n

h=0.25 .\* h;

y=(exp(x + h) - exp(x))/h;

error = abs(exp(x) - y);

fprintf('%d %f %f %f\n',i,h,y,err);

if error > emax

emax=error;

imax=i;

end

end

disp('imax and emax values are');

fprintf('%d %f\n',imax,emax);

Output:

1 0.250000 1.873115 0.877583  
2 0.062500 1.701334 0.877583  
3 0.015625 1.661669 0.877583  
4 0.003906 1.651946 0.877583  
5 0.000977 1.649527 0.877583  
6 0.000244 1.648923 0.877583  
7 0.000061 1.648772 0.877583  
8 0.000015 1.648734 0.877583  
9 0.000004 1.648724 0.877583  
10 0.000001 1.648722 0.877583  
11 0.000000 1.648721 0.877583  
12 0.000000 1.648721 0.877583  
13 0.000000 1.648721 0.877583  
14 0.000000 1.648721 0.877583  
15 0.000000 1.648721 0.877583  
16 0.000000 1.648721 0.877583  
17 0.000000 1.648720 0.877583  
18 0.000000 1.648712 0.877583  
19 0.000000 1.648682 0.877583  
20 0.000000 1.648682 0.877583  
21 0.000000 1.648438 0.877583  
22 0.000000 1.648438 0.877583  
23 0.000000 1.640625 0.877583  
24 0.000000 1.625000 0.877583  
25 0.000000 1.500000 0.877583  
26 0.000000 2.000000 0.877583  
27 0.000000 0.000000 0.877583  
28 0.000000 0.000000 0.877583  
29 0.000000 0.000000 0.877583  
30 0.000000 0.000000 0.877583  
imax and emax values are  
27 1.648721

21. Write a computer code that contains the following assignment statements exactly as shown. Analyze the results.

a. Print these values first using the default format and then with an extremely large format field:

**real** p, q, u, v,w, x, y, z

x ← 0.1

y ← 0.01

z ← x − y

p ← 1.0/3.0

q ← 3.0p

u ← 7.6

v ← 2.9

w ← u − v

**output** x, y, z, p, q, u, v,w

Answer:

Code:

Mathlab

format

x=0.1;

y=0.01;

z=x-y;

p=1.0/3.0;

q=3.0\*p;

u=7.6;

v=2.9;

w=u-v;

disp('The given data in default format');

%Enable defalut format

format

fprintf('x =');disp(x)

fprintf('y =');disp(y)

fprintf('z =');disp(z)

fprintf('p =');disp(p)

fprintf('q =');disp(q)

fprintf('u =');disp(u)

fprintf('v =');disp(v)

fprintf('w =');disp(w)

disp('The given data in extreamly large format');

%Enable long format

format long

fprintf('x =');disp(x)

fprintf('y =');disp(y)

fprintf('z =');disp(z)

fprintf('p =');disp(p)

fprintf('q =');disp(q)

fprintf('u =');disp(u)

fprintf('v =');disp(v)

fprintf('w =');disp(w)

Output:

The given data in default format  
x = 0.1000  
  
y = 0.0100  
  
z = 0.0900  
  
p = 0.3333  
  
q = 1  
  
u = 7.6000  
  
v = 2.9000  
  
w = 4.7000  
  
The given data in extreamly large format  
x = 0.100000000000000  
  
y = 0.010000000000000  
  
z = 0.090000000000000  
  
p = 0.333333333333333  
  
q = 1  
  
u = 7.600000000000000  
  
v = 2.900000000000000  
  
w = 4.699999999999999

b. What values would be computed for x, y, and z if this code is used?

**integer** n; **real** x, y, z

**for** n = 1 to 10 **do**

x ← (n − 1)/2

y ← n2/3.0

z ← 1.0 + 1/n

**output** x, y, z

**end for**

**Answer:**

Code: MathLab

for n = 1:10

x = (n - 1)/2;

y = n^2 / 3.0;

z = 1.0 + 1/n;

end

disp("x = " + num2str(x))

disp("y = " + num2str(y))

disp("z = " + num2str(z))

Output:

x = 4.5  
y = 33.3333  
z = 1.1

Section 2.2: 6

Write a procedure to compute f (x) = sin x − 1 + cos x. The routine should produce nearly full machine precision for all x in the interval [0, π/4]. Hint: The trigonometric identity sin2 θ = 1/2 (1 − cos 2θ) may be useful.

**Answer:**

clc;

% create vector for value of x in range 0 to pi/4 with each step of 0.001

x = [ 0 : 0.001 : pi / 4 ];

% create vector f of same length as x

% store the value of f(x) = sin(x)-1+cos(x)

f = [1 : length(x)];

fprintf('%5s %10s\n\n', 'x', 'f(x)');

% find the value of f(x) for each x

for i = 1 : length(x)

f(i) = sin( x(i) )-1+cos( x(i) );

fprintf('%5.3f %10.3f\n', x(i), f(i));

end

% Display diagram

xlabel('x');

ylabel('f(x) = sin(x)-1+cos(x)');

hold

plot(x, f);

Output:

x f(x)  
  
0.000 0.000  
0.001 0.001  
0.002 0.002  
0.003 0.003  
0.004 0.004  
0.005 0.005  
0.006 0.006  
0.007 0.007  
0.008 0.008  
0.009 0.009  
0.010 0.010  
0.011 0.011  
0.012 0.012  
0.013 0.013  
0.014 0.014  
0.015 0.015  
0.016 0.016  
0.017 0.017  
0.018 0.018  
0.019 0.019  
0.020 0.020  
0.021 0.021  
0.022 0.022  
0.023 0.023  
0.024 0.024  
0.025 0.025  
0.026 0.026  
0.027 0.027  
0.028 0.028  
0.029 0.029  
0.030 0.030  
0.031 0.031  
0.032 0.031  
0.033 0.032  
0.034 0.033  
0.035 0.034  
0.036 0.035  
0.037 0.036  
0.038 0.037  
0.039 0.038  
0.040 0.039  
0.041 0.040  
0.042 0.041  
0.043 0.042  
0.044 0.043  
0.045 0.044  
0.046 0.045  
0.047 0.046  
0.048 0.047  
0.049 0.048  
0.050 0.049  
0.051 0.050  
0.052 0.051  
0.053 0.052  
0.054 0.053  
0.055 0.053  
0.056 0.054  
0.057 0.055  
0.058 0.056  
0.059 0.057  
0.060 0.058  
0.061 0.059  
0.062 0.060  
0.063 0.061  
0.064 0.062  
0.065 0.063  
0.066 0.064  
0.067 0.065  
0.068 0.066  
0.069 0.067  
0.070 0.067  
0.071 0.068  
0.072 0.069  
0.073 0.070  
0.074 0.071  
0.075 0.072  
0.076 0.073  
0.077 0.074  
0.078 0.075  
0.079 0.076  
0.080 0.077  
0.081 0.078  
0.082 0.079  
0.083 0.079  
0.084 0.080  
0.085 0.081  
0.086 0.082  
0.087 0.083  
0.088 0.084  
0.089 0.085  
0.090 0.086  
0.091 0.087  
0.092 0.088  
0.093 0.089  
0.094 0.089  
0.095 0.090  
0.096 0.091  
0.097 0.092  
0.098 0.093  
0.099 0.094  
0.100 0.095  
0.101 0.096  
0.102 0.097  
0.103 0.098  
0.104 0.098  
0.105 0.099  
0.106 0.100  
0.107 0.101  
0.108 0.102  
0.109 0.103  
0.110 0.104  
0.111 0.105  
0.112 0.106  
0.113 0.106  
0.114 0.107  
0.115 0.108  
0.116 0.109  
0.117 0.110  
0.118 0.111  
0.119 0.112  
0.120 0.113  
0.121 0.113  
0.122 0.114  
0.123 0.115  
0.124 0.116  
0.125 0.117  
0.126 0.118  
0.127 0.119  
0.128 0.119  
0.129 0.120  
0.130 0.121  
0.131 0.122  
0.132 0.123  
0.133 0.124  
0.134 0.125  
0.135 0.125  
0.136 0.126  
0.137 0.127  
0.138 0.128  
0.139 0.129  
0.140 0.130  
0.141 0.131  
0.142 0.131  
0.143 0.132  
0.144 0.133  
0.145 0.134  
0.146 0.135  
0.147 0.136  
0.148 0.137  
0.149 0.137  
0.150 0.138  
0.151 0.139  
0.152 0.140  
0.153 0.141  
0.154 0.142  
0.155 0.142  
0.156 0.143  
0.157 0.144  
0.158 0.145  
0.159 0.146  
0.160 0.147  
0.161 0.147  
0.162 0.148  
0.163 0.149  
0.164 0.150  
0.165 0.151  
0.166 0.151  
0.167 0.152  
0.168 0.153  
0.169 0.154  
0.170 0.155  
0.171 0.156  
0.172 0.156  
0.173 0.157  
0.174 0.158  
0.175 0.159  
0.176 0.160  
0.177 0.160  
0.178 0.161  
0.179 0.162  
0.180 0.163  
0.181 0.164  
0.182 0.164  
0.183 0.165  
0.184 0.166  
0.185 0.167  
0.186 0.168  
0.187 0.168  
0.188 0.169  
0.189 0.170  
0.190 0.171  
0.191 0.172  
0.192 0.172  
0.193 0.173  
0.194 0.174  
0.195 0.175  
0.196 0.176  
0.197 0.176

0.198 0.177  
0.199 0.178  
0.200 0.179  
0.201 0.180  
0.202 0.180  
0.203 0.181  
0.204 0.182  
0.205 0.183  
0.206 0.183  
0.207 0.184  
0.208 0.185  
0.209 0.186  
0.210 0.186  
0.211 0.187  
0.212 0.188  
0.213 0.189  
0.214 0.190  
0.215 0.190  
0.216 0.191  
0.217 0.192  
0.218 0.193  
0.219 0.193  
0.220 0.194  
0.221 0.195  
0.222 0.196  
0.223 0.196  
0.224 0.197  
0.225 0.198  
0.226 0.199  
0.227 0.199  
0.228 0.200  
0.229 0.201  
0.230 0.202  
0.231 0.202  
0.232 0.203  
0.233 0.204  
0.234 0.205  
0.235 0.205  
0.236 0.206  
0.237 0.207  
0.238 0.208  
0.239 0.208  
0.240 0.209  
0.241 0.210  
0.242 0.211  
0.243 0.211  
0.244 0.212  
0.245 0.213  
0.246 0.213  
0.247 0.214  
0.248 0.215  
0.249 0.216  
0.250 0.216  
0.251 0.217  
0.252 0.218  
0.253 0.218  
0.254 0.219  
0.255 0.220  
0.256 0.221  
0.257 0.221  
0.258 0.222  
0.259 0.223  
0.260 0.223  
0.261 0.224  
0.262 0.225  
0.263 0.226  
0.264 0.226  
0.265 0.227  
0.266 0.228  
0.267 0.228  
0.268 0.229  
0.269 0.230  
0.270 0.231  
0.271 0.231  
0.272 0.232  
0.273 0.233  
0.274 0.233  
0.275 0.234  
0.276 0.235  
0.277 0.235  
0.278 0.236  
0.279 0.237  
0.280 0.237  
0.281 0.238  
0.282 0.239  
0.283 0.239  
0.284 0.240  
0.285 0.241  
0.286 0.241  
0.287 0.242  
0.288 0.243  
0.289 0.244  
0.290 0.244  
0.291 0.245  
0.292 0.246  
0.293 0.246  
0.294 0.247  
0.295 0.248  
0.296 0.248  
0.297 0.249  
0.298 0.250  
0.299 0.250  
0.300 0.251  
0.301 0.252  
0.302 0.252  
0.303 0.253  
0.304 0.253  
0.305 0.254  
0.306 0.255  
0.307 0.255  
0.308 0.256  
0.309 0.257  
0.310 0.257  
0.311 0.258  
0.312 0.259  
0.313 0.259  
0.314 0.260  
0.315 0.261  
0.316 0.261  
0.317 0.262  
0.318 0.263  
0.319 0.263  
0.320 0.264  
0.321 0.264  
0.322 0.265  
0.323 0.266  
0.324 0.266  
0.325 0.267  
0.326 0.268  
0.327 0.268  
0.328 0.269  
0.329 0.269  
0.330 0.270  
0.331 0.271  
0.332 0.271  
0.333 0.272  
0.334 0.273  
0.335 0.273  
0.336 0.274  
0.337 0.274  
0.338 0.275  
0.339 0.276  
0.340 0.276  
0.341 0.277  
0.342 0.277  
0.343 0.278  
0.344 0.279  
0.345 0.279  
0.346 0.280  
0.347 0.280  
0.348 0.281  
0.349 0.282  
0.350 0.282  
0.351 0.283  
0.352 0.283  
0.353 0.284  
0.354 0.285  
0.355 0.285  
0.356 0.286  
0.357 0.286  
0.358 0.287  
0.359 0.288  
0.360 0.288  
0.361 0.289  
0.362 0.289  
0.363 0.290  
0.364 0.290  
0.365 0.291  
0.366 0.292  
0.367 0.292  
0.368 0.293  
0.369 0.293  
0.370 0.294  
0.371 0.295  
0.372 0.295  
0.373 0.296  
0.374 0.296  
0.375 0.297  
0.376 0.297  
0.377 0.298  
0.378 0.298  
0.379 0.299  
0.380 0.300  
0.381 0.300  
0.382 0.301  
0.383 0.301  
0.384 0.302  
0.385 0.302  
0.386 0.303  
0.387 0.303  
0.388 0.304  
0.389 0.305  
0.390 0.305  
0.391 0.306  
0.392 0.306  
0.393 0.307  
0.394 0.307  
0.395 0.308  
0.396 0.308  
0.397 0.309

0.398 0.309  
0.399 0.310  
0.400 0.310  
0.401 0.311  
0.402 0.312  
0.403 0.312  
0.404 0.313  
0.405 0.313  
0.406 0.314  
0.407 0.314  
0.408 0.315  
0.409 0.315  
0.410 0.316  
0.411 0.316  
0.412 0.317  
0.413 0.317  
0.414 0.318  
0.415 0.318  
0.416 0.319  
0.417 0.319  
0.418 0.320  
0.419 0.320  
0.420 0.321  
0.421 0.321  
0.422 0.322  
0.423 0.322  
0.424 0.323  
0.425 0.323  
0.426 0.324  
0.427 0.324  
0.428 0.325  
0.429 0.325  
0.430 0.326  
0.431 0.326  
0.432 0.327  
0.433 0.327  
0.434 0.328  
0.435 0.328  
0.436 0.329  
0.437 0.329  
0.438 0.330  
0.439 0.330  
0.440 0.331  
0.441 0.331  
0.442 0.332  
0.443 0.332  
0.444 0.333  
0.445 0.333  
0.446 0.334  
0.447 0.334  
0.448 0.334  
0.449 0.335  
0.450 0.335  
0.451 0.336  
0.452 0.336  
0.453 0.337  
0.454 0.337  
0.455 0.338  
0.456 0.338  
0.457 0.339  
0.458 0.339  
0.459 0.340  
0.460 0.340  
0.461 0.340  
0.462 0.341  
0.463 0.341  
0.464 0.342  
0.465 0.342  
0.466 0.343  
0.467 0.343  
0.468 0.344  
0.469 0.344  
0.470 0.344  
0.471 0.345  
0.472 0.345  
0.473 0.346  
0.474 0.346  
0.475 0.347  
0.476 0.347  
0.477 0.347  
0.478 0.348  
0.479 0.348  
0.480 0.349  
0.481 0.349  
0.482 0.350  
0.483 0.350  
0.484 0.350  
0.485 0.351  
0.486 0.351  
0.487 0.352  
0.488 0.352  
0.489 0.353  
0.490 0.353  
0.491 0.353  
0.492 0.354  
0.493 0.354  
0.494 0.355  
0.495 0.355  
0.496 0.355  
0.497 0.356  
0.498 0.356  
0.499 0.357  
0.500 0.357  
0.501 0.357  
0.502 0.358  
0.503 0.358  
0.504 0.359  
0.505 0.359  
0.506 0.359  
0.507 0.360  
0.508 0.360  
0.509 0.361  
0.510 0.361  
0.511 0.361  
0.512 0.362  
0.513 0.362  
0.514 0.362  
0.515 0.363  
0.516 0.363  
0.517 0.364  
0.518 0.364  
0.519 0.364  
0.520 0.365  
0.521 0.365  
0.522 0.365  
0.523 0.366  
0.524 0.366  
0.525 0.367  
0.526 0.367  
0.527 0.367  
0.528 0.368  
0.529 0.368  
0.530 0.368  
0.531 0.369  
0.532 0.369  
0.533 0.369  
0.534 0.370  
0.535 0.370  
0.536 0.370  
0.537 0.371  
0.538 0.371  
0.539 0.372  
0.540 0.372  
0.541 0.372  
0.542 0.373  
0.543 0.373  
0.544 0.373  
0.545 0.374  
0.546 0.374  
0.547 0.374  
0.548 0.375  
0.549 0.375  
0.550 0.375  
0.551 0.376  
0.552 0.376  
0.553 0.376  
0.554 0.377  
0.555 0.377  
0.556 0.377  
0.557 0.377  
0.558 0.378  
0.559 0.378  
0.560 0.378  
0.561 0.379  
0.562 0.379  
0.563 0.379  
0.564 0.380  
0.565 0.380  
0.566 0.380  
0.567 0.381  
0.568 0.381  
0.569 0.381  
0.570 0.382  
0.571 0.382  
0.572 0.382  
0.573 0.382  
0.574 0.383  
0.575 0.383  
0.576 0.383  
0.577 0.384  
0.578 0.384  
0.579 0.384  
0.580 0.384  
0.581 0.385  
0.582 0.385  
0.583 0.385  
0.584 0.386  
0.585 0.386  
0.586 0.386  
0.587 0.386  
0.588 0.387  
0.589 0.387  
0.590 0.387  
0.591 0.388  
0.592 0.388  
0.593 0.388  
0.594 0.388  
0.595 0.389  
0.596 0.389  
0.597 0.389

0.598 0.389  
0.599 0.390  
0.600 0.390  
0.601 0.390  
0.602 0.390  
0.603 0.391  
0.604 0.391  
0.605 0.391  
0.606 0.392  
0.607 0.392  
0.608 0.392  
0.609 0.392  
0.610 0.393  
0.611 0.393  
0.612 0.393  
0.613 0.393  
0.614 0.393  
0.615 0.394  
0.616 0.394  
0.617 0.394  
0.618 0.394  
0.619 0.395  
0.620 0.395  
0.621 0.395  
0.622 0.395  
0.623 0.396  
0.624 0.396  
0.625 0.396  
0.626 0.396  
0.627 0.397  
0.628 0.397  
0.629 0.397  
0.630 0.397  
0.631 0.397  
0.632 0.398  
0.633 0.398  
0.634 0.398  
0.635 0.398  
0.636 0.398  
0.637 0.399  
0.638 0.399  
0.639 0.399  
0.640 0.399  
0.641 0.399  
0.642 0.400  
0.643 0.400  
0.644 0.400  
0.645 0.400  
0.646 0.400  
0.647 0.401  
0.648 0.401  
0.649 0.401  
0.650 0.401  
0.651 0.401  
0.652 0.402  
0.653 0.402  
0.654 0.402  
0.655 0.402  
0.656 0.402  
0.657 0.403  
0.658 0.403  
0.659 0.403  
0.660 0.403  
0.661 0.403  
0.662 0.403  
0.663 0.404  
0.664 0.404  
0.665 0.404  
0.666 0.404  
0.667 0.404  
0.668 0.404  
0.669 0.405  
0.670 0.405  
0.671 0.405  
0.672 0.405  
0.673 0.405  
0.674 0.405  
0.675 0.406  
0.676 0.406  
0.677 0.406  
0.678 0.406  
0.679 0.406  
0.680 0.406  
0.681 0.407  
0.682 0.407  
0.683 0.407  
0.684 0.407  
0.685 0.407  
0.686 0.407  
0.687 0.407  
0.688 0.408  
0.689 0.408  
0.690 0.408  
0.691 0.408  
0.692 0.408  
0.693 0.408  
0.694 0.408  
0.695 0.408  
0.696 0.409  
0.697 0.409  
0.698 0.409  
0.699 0.409  
0.700 0.409  
0.701 0.409  
0.702 0.409  
0.703 0.409  
0.704 0.410  
0.705 0.410  
0.706 0.410  
0.707 0.410  
0.708 0.410  
0.709 0.410  
0.710 0.410  
0.711 0.410  
0.712 0.410  
0.713 0.411  
0.714 0.411  
0.715 0.411  
0.716 0.411  
0.717 0.411  
0.718 0.411  
0.719 0.411  
0.720 0.411  
0.721 0.411  
0.722 0.411  
0.723 0.411  
0.724 0.412  
0.725 0.412  
0.726 0.412  
0.727 0.412  
0.728 0.412  
0.729 0.412  
0.730 0.412  
0.731 0.412  
0.732 0.412  
0.733 0.412  
0.734 0.412  
0.735 0.412  
0.736 0.412  
0.737 0.413  
0.738 0.413  
0.739 0.413  
0.740 0.413  
0.741 0.413  
0.742 0.413  
0.743 0.413  
0.744 0.413  
0.745 0.413  
0.746 0.413  
0.747 0.413  
0.748 0.413  
0.749 0.413  
0.750 0.413  
0.751 0.413  
0.752 0.413  
0.753 0.413  
0.754 0.414  
0.755 0.414  
0.756 0.414  
0.757 0.414  
0.758 0.414  
0.759 0.414  
0.760 0.414  
0.761 0.414  
0.762 0.414  
0.763 0.414  
0.764 0.414  
0.765 0.414  
0.766 0.414  
0.767 0.414  
0.768 0.414  
0.769 0.414  
0.770 0.414  
0.771 0.414  
0.772 0.414  
0.773 0.414  
0.774 0.414  
0.775 0.414  
0.776 0.414  
0.777 0.414  
0.778 0.414  
0.779 0.414  
0.780 0.414  
0.781 0.414  
0.782 0.414  
0.783 0.414  
0.784 0.414  
0.785 0.414  
Current plot released

Chart

Description automatically generated