Original problem in HW1 is a stable iteration

```
> restart:
> ser:=convert(series( (sqrt(1+(t+dt)^2)-1)/(t+dt), dt, 2),
  polynom);
```

$$ser := \frac{\sqrt{t^2+1}-1}{t} + \frac{\left(\frac{t}{\sqrt{t^2+1}} - \frac{\sqrt{t^2+1}-1}{t}\right) dt}{t} \quad (1)$$

```
> x:=eval(ser,dt=0);
```

$$x := \frac{\sqrt{t^2+1}-1}{t} \quad (2)$$

```
> dx:=simplify(ser-x);
```

$$dx := \frac{\left(\sqrt{t^2+1}-1\right) dt}{\sqrt{t^2+1}\, t^2} \quad (3)$$

```
> dt:=t*rel_t;
```

$$dt := t\, rel\_t \quad (4)$$

```
> rel_x:=dx/x;
```

$$rel\_x := \frac{rel\_t}{\sqrt{t^2+1}} \quad (5)$$

We can cause a problem with error propagation (is it serious or not?) by changing slightly the iteration:

```
> restart:
> ser:=convert(series( (1-sqrt(1-(t+dt)^2))/(t+dt), dt, 2),
  polynom);
```

$$ser := \frac{1-\sqrt{-t^2+1}}{t} + \frac{\left(\frac{t}{\sqrt{-t^2+1}} + \frac{-1+\sqrt{-t^2+1}}{t}\right) dt}{t} \quad (6)$$

```
> x:=eval(ser,dt=0):
> dx:=simplify(ser-x):
> dt:=t*rel_t:
> rel_x:=simplify(dx/x);
```

$$rel\_x := \frac{rel\_t}{\sqrt{-t^2+1}} \quad (7)$$

Original iteration with roundoff problem and 100 digits

```
> restart:
> N_iter:=25:
> x:=Vector(N_iter,0):
> Digits:=100:
> t[0]:=1/sqrt(3.0);
```

$$t_0 := 0.57735026918962576450914878050195745564760175127012687601860232648397767230293334569371539558574952 51 \quad (8)$$

```
> for i from 1 to N_iter do; t[i]:=(sqrt(1+t[i-1]^2)-1)/t[i-1]: x
```

```
  [i]:=6*2^i*t[i]: od:
> err:=(x[N_iter]-Pi)/Pi: evalhf(err);
```

$$8.11663855557865576\ 10^{-17} \tag{9}$$

```
> evalhf(x[N_iter-9. .N_iter]) ;
```

$$\begin{bmatrix} 3.14159265365663787 \\ 3.14159265360650419 \\ 3.14159265359397111 \\ 3.14159265359083761 \\ 3.14159265359005424 \\ 3.14159265358985840 \\ 3.14159265358980955 \\ 3.14159265358979711 \\ 3.14159265358979445 \\ 3.14159265358979356 \end{bmatrix} \tag{10}$$

Original iteration with roundoff problem and 16 digits

```
> restart:
> N_iter:=20:
> x:=Vector(N_iter,0):
> Digits:=16:
> t[0]:=1/sqrt(3.0);
```

$$t_0 := 0.5773502691896259 \tag{11}$$

```
> for i from 1 to N_iter do; t[i]:=(sqrt(1+t[i-1]^2)-1)/t[i-1]: x
  [i]:=6*2^i*t[i]: od:
> err:=(x[N_iter]-Pi)/Pi: evalhf(err);
```

$$-0.00110083164127062799 \tag{12}$$

```
> evalhf(x[N_iter-9..N_iter]);
```

$$\begin{bmatrix} 3.14159270977275407 \\ 3.14159263799051303 \\ 3.14159270977275407 \\ 3.14159263799051303 \\ 3.14158809570712094 \\ 3.14158033379770218 \\ 3.14151427036785202 \\ 3.14128502623068417 \\ 3.14072672605339598 \\ 3.13813428899273816 \end{bmatrix} \tag{13}$$

Fixed iteration with 16 digits

```
> restart:
> N_iter:=25:
```

```
> x:=Vector(N_iter,0):
> Digits:=16:
> t[0]:=1/sqrt(3.0);
```

$$t_0 := 0.5773502691896259 \tag{14}$$

```
> for i from 1 to N_iter do; t[i]:=t[i-1]/(sqrt(1+t[i-1]^2)+1): x
  [i]:=6*2^i*t[i]: od:
> err:=(x[N_iter]-Pi)/Pi: evalhf(err);
```

$$2.22816920328653517\ 10^{-15} \tag{15}$$

```
> evalhf(x[N_iter-9..N_iter]);
```

$$\begin{bmatrix} 3.14159265365664186 \\ 3.14159265360650819 \\ 3.14159265359397422 \\ 3.14159265359084205 \\ 3.14159265359006001 \\ 3.14159265358986506 \\ 3.14159265358981621 \\ 3.14159265358980422 \\ 3.14159265358980022 \\ 3.14159265358980022 \end{bmatrix} \tag{16}$$

New iteration with stability problem and 100 digits

```
> restart:
> N_iter:=25:
> x:=Vector(N_iter,0):
> Digits:=100:
> t[0]:=0.99999;
```

$$t_0 := 0.99999 \tag{17}$$

```
> t0:=t[0]:
> for i from 1 to N_iter do; t[i]:=t[i-1]/(1+sqrt(1-t[i-1]^2)): x
  [i]:=2^i*t[i]: od:
> x_true:=convert(x,list): t_true:=[seq(t[i],i=1..N_iter)]:
> evalhf(x[N_iter-9..N_iter]);
```

(18)

$$\begin{bmatrix} 6.10303380511649607 \\ 6.10303381834825132 \\ 6.10303382165619013 \\ 6.10303382248317483 \\ 6.10303382268992145 \\ 6.10303382274160811 \\ 6.10303382275452933 \\ 6.10303382275775963 \\ 6.10303382275856787 \\ 6.10303382275876949 \end{bmatrix} \quad (18)$$

This iteration also converges to some number:

```
> x_true[N_iter];
```

$$6.10303382275876954372984493443240375780958444950709968002913112362224675600911\backslash \quad (19)$$
$$6758726670068313643863 1$$

Although this iteration has some initial growth of the errors, it is not catastrophic, but we do loose almost 4 digits.

New iteration with stability problem and 16 digits

```
> x:=Vector(N_iter,0):
> Digits:=16:
> dt:=1e-12: # Perturb the initial iteration by a little
> t[0]:=t0+dt;
```

$$t_0 := 0.999990000001 \quad (20)$$

```
> for i from 1 to N_iter do; t[i]:=t[i-1]/(1+sqrt(1-t[i-1]^2)): x
  [i]:=2^i*t[i]: od:
> rel_err:=(x[N_iter]-x_true[N_iter])/x_true[N_iter];
```

$$rel\_err := 8.192780746772595\ 10^{-9} \quad (21)$$

```
> rel_err_1:=(x[1]-x_true[1])/x_true[1];
```

$$rel\_err\_1 := 2.236117937025200\ 10^{-10} \quad (22)$$

```
> Digits:=12: # Just to print shorter numbers (there are better
  ways)
> seq((x[i]-x_true[i])/x_true[i], i=1..10);
```

$$2.20986077311\ 10^{-10}, 2.37167156803\ 10^{-9}, 5.70577561816\ 10^{-9}, 7.44936825550\ 10^{-9}, \quad (23)$$
$$7.99821230998\ 10^{-9}, 8.14350779667\ 10^{-9}, 8.17917685155\ 10^{-9}, 8.18928221879\ 10^{-9},$$
$$8.19139612489\ 10^{-9}, 8.19274371995\ 10^{-9}$$

```
> rel_err_1/dt;
```

$$223.611793703 \quad (24)$$

```
> evalf(1/sqrt(1-t0^2));
```

$$223.607356769 \quad (25)$$

How many digits did we perturb the answer by:

```
> rel_err/dt;
```

$$8192.78074677 \quad (26)$$

```
> mul(1/sqrt(1-t[i]^2), i=0..N_iter);
                                8192.60650364
```
(27)