Ejercicio 1

Ejemplo de estimación del error en el método de Newton por cota inferior de la derivada.

El error se estima como:
$$
|x_n-α| ≤ $\frac{f(x\_n)}{c}$
$$
con c el ínfimo de la derivada. Intervalo [0,0.5]. En 6 iteraciones el error es del orden de 10^-9.

Para introducir las derivadas de f como funciones que evalúan numéricamente pueden usarse los comandos $D(f)$ y $D[1,1](f)$ para la primera y para la segunda derivada.

Ver al final para la derivada segunda.

Debajo, el intervalo es [a1,b1].

Las iteradas del método de Newton se designan por a[i]; donde a[1] es el dato inicial.

```
> unassign('f');
> f := x→x^6 - x - 1;
```

$$f := x \to x^6 - x - 1 \quad (1)$$

```
>
> a1 := 1; b1 := 2;
```

$$a1 := 1$$
$$b1 := 2 \quad (2)$$

```
> unassign('g');
> g := D(f);
```

$$g := x \to 6\,x^5 - 1 \quad (3)$$

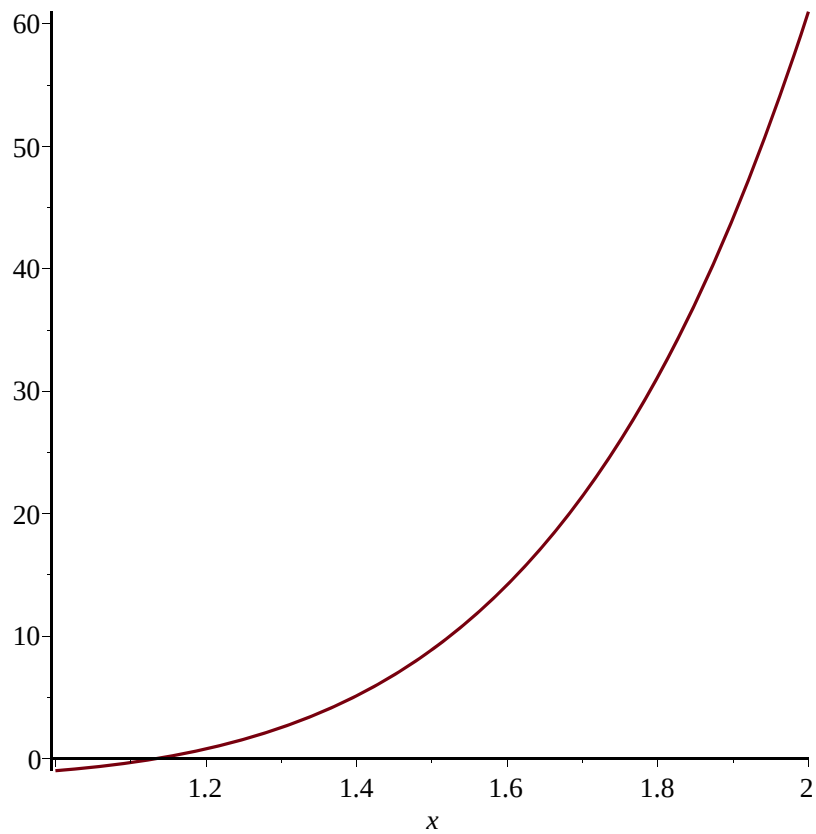
```
> g(x);
```

$$6\,x^5 - 1 \quad (4)$$

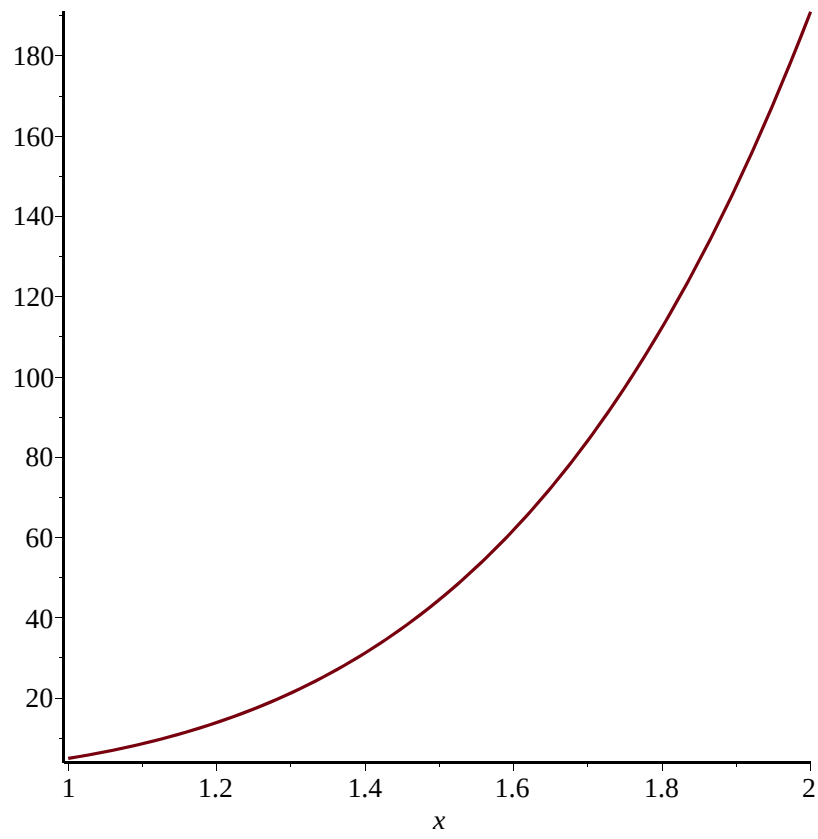
```
> f(a1)/g(a1); evalf(f(b1)/g(b1));
```

$$-\frac{1}{5}$$
$$0.3193717277 \quad (5)$$

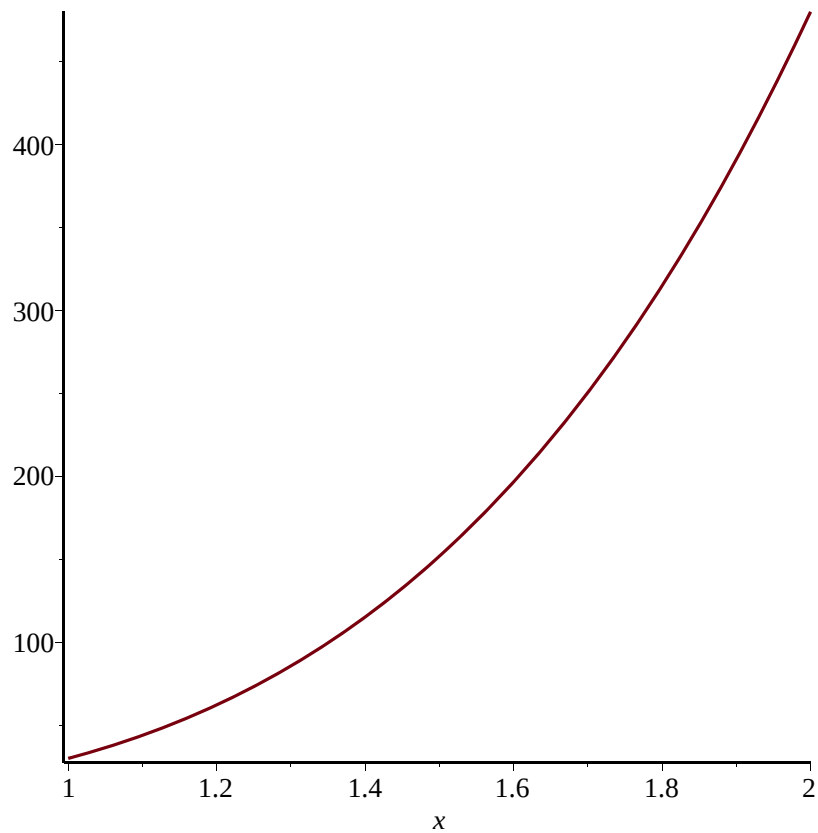
```
> unassign('h');
> plot(f(x), x = a1 ..b1);
```



> $plot(diff(f(x), x), x = a1..b1);$



> $plot(diff(f(x), x\$2), x = a1..b1);$



> $unassign('a'); unassign('b');$

> $a[1] := 1;$ **for** i **from** 2 **to** 8 **do** $a[i] := evalf\left(a[i-1] - \frac{f(a[i-1])}{g(a[i-1])}\right);$ **end do**;

$$a_1 := 1$$
$$a_2 := 1.200000000$$
$$a_3 := 1.143575842$$
$$a_4 := 1.134909462$$
$$a_5 := 1.134724221$$
$$a_6 := 1.134724138$$
$$a_7 := 1.134724138$$
$$a_8 := 1.134724138 \quad \textbf{(6)}$$

> **for** i **from** 2 **to** 8 **do** $print\left(i, \frac{f(a[i])}{5}\right);$ **end do**;

2, 0.1571968000
3, 0.01860639040
4, 0.0003814790000

$$5,\ 1.700000000\ 10^{-7}$$
$$6,\ -8.000000000\ 10^{-10}$$
$$7,\ -8.000000000\ 10^{-10}$$
$$8,\ -8.000000000\ 10^{-10} \tag{7}$$

>

Hemos usado arriba que la cota inferior de la derivada es 5 por eso dividimos por 5.

?*diff*

D[1, 1](f) (x);

$$12\,x \tag{8}$$

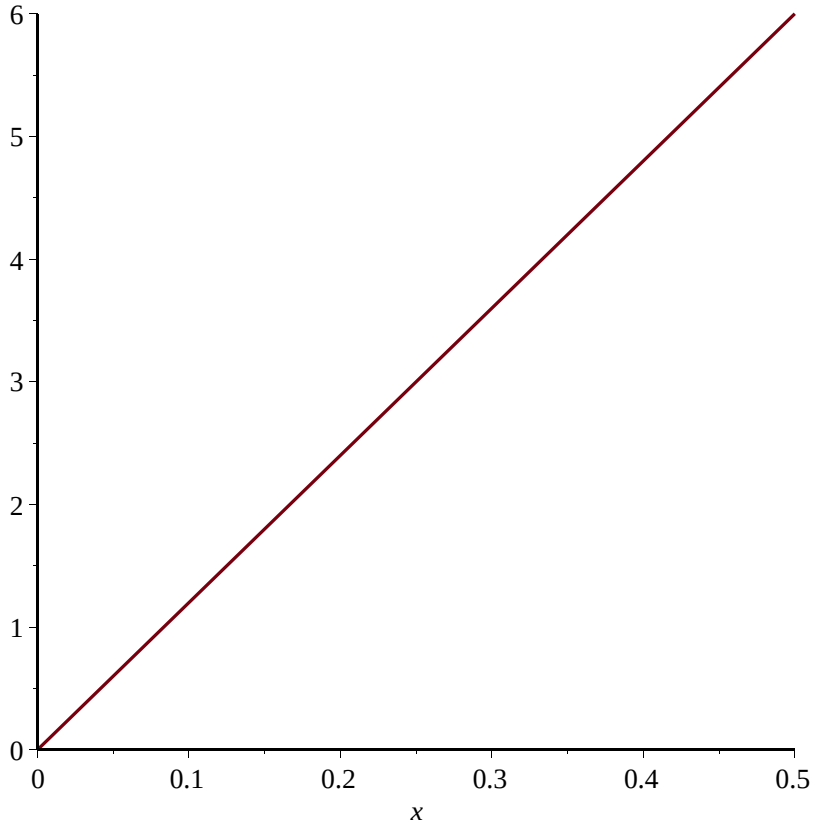
h := D[1, 1](f);

$$x \rightarrow 12\,x \tag{9}$$

$h(5)$;

$$60 \tag{10}$$

$plot(h(x), x = 0\,..0.5)$;



Ejercicio 2

Ejemplo de estimación del error en el método de Newton por cota inferior de la derivada.

El error se estima como:
$$
|x_n-α| ≤ $\frac{f(x\_n)}{c}$
$$
con c el ínfimo de la derivada.

Para introducir las derivadas de f como funciones que evalúan numéricamente pueden usarse los comandos $D(f)$ y $D[1,1](f)$ para la primera y para la segunda derivada.

Ver al final para la derivada segunda.

Debajo, el interval es [a1,b1].

Las iteradas del método de Newton se designan por a[i]; donde a[1] es el dato inicial.

```
> unassign('f');
> f := x→x − 1 − ln(x + 1);
```

$$f := x \to x - 1 - \ln(x+1) \quad (1)$$

```
>
> a1 := 1; b1 := 3;
```

$$a1 := 1$$
$$b1 := 3 \quad (2)$$

```
> unassign('g');
> g := D(f);
```

$$g := x \to 1 - \frac{1}{x+1} \quad (3)$$

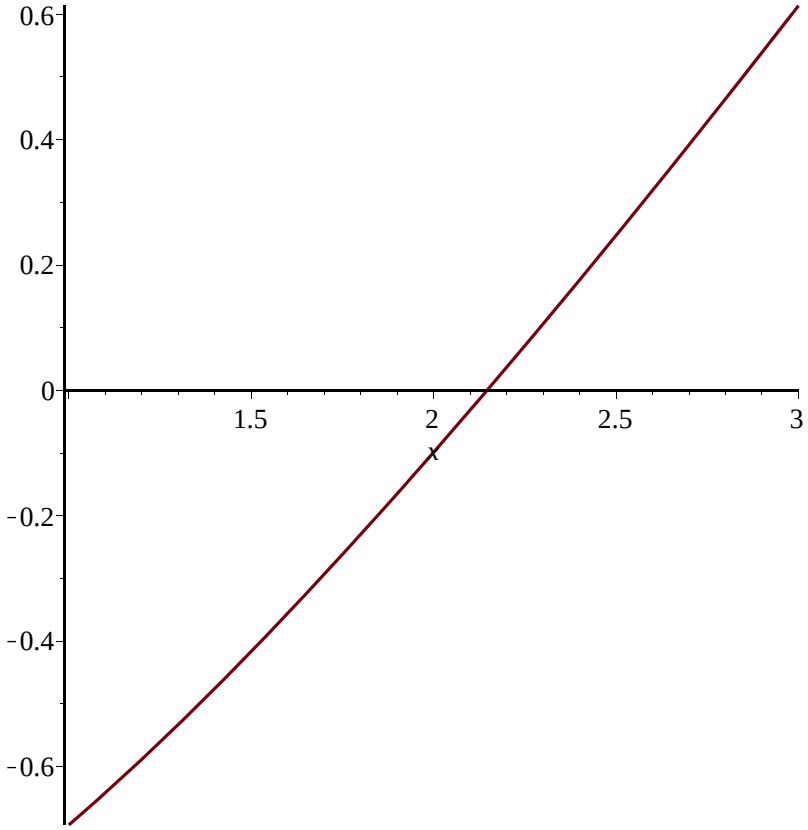
```
> g(x);
```

$$1 - \frac{1}{x+1} \quad (4)$$

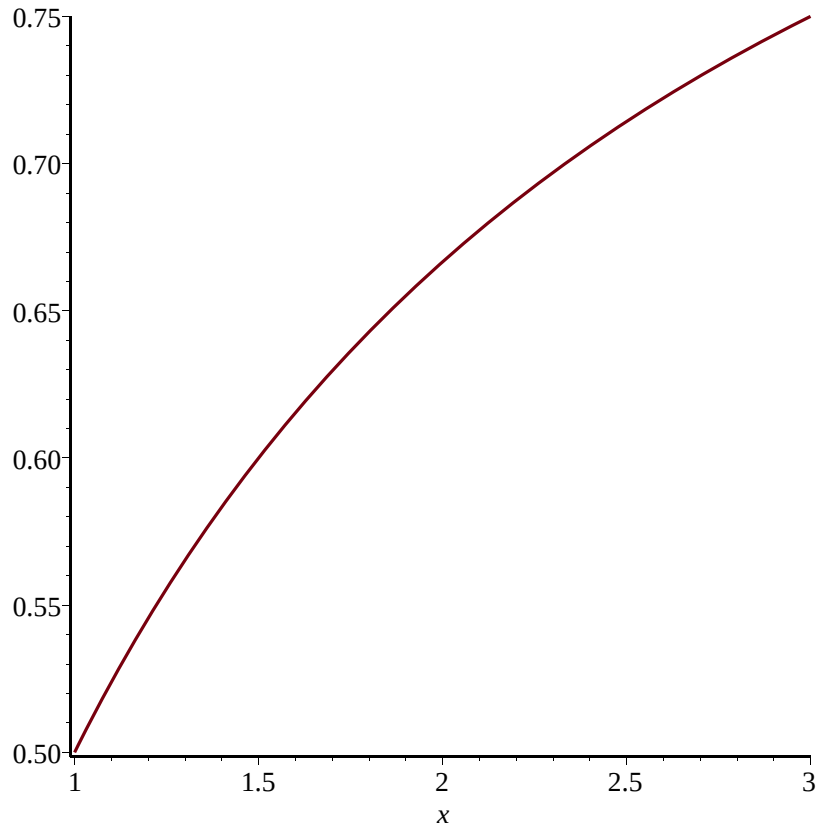
```
> evalf(f(a1)/g(a1)); evalf(f(b1)/g(b1));
```

$$-1.386294361$$
$$0.818274185 \quad (5)$$

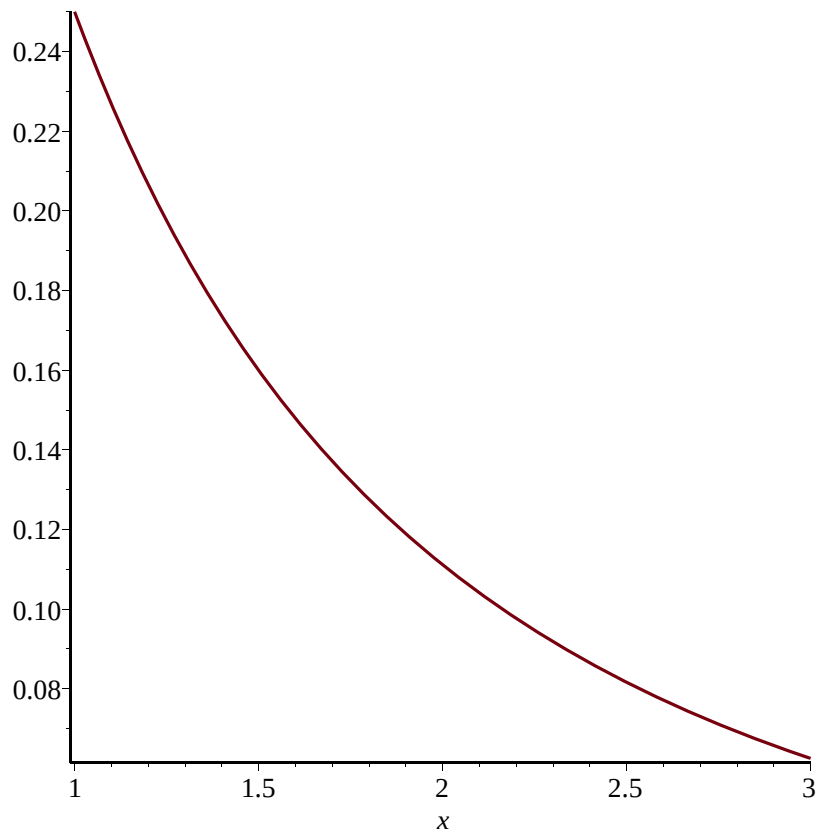
```
> unassign('h');
> plot(f(x), x = a1 ..b1);
```



> $plot(diff(f(x), x), x = a1 .. b1);$



> $plot(diff(f(x), x\$2), x = a1..b1);$



> $unassign('a'); unassign('b');$

> $a[1] := 1;$ **for** i **from** 2 **to** 8 **do** $a[i] := evalf\left(a[i-1] - \frac{f(a[i-1])}{g(a[i-1])}\right);$ **end do**;

$$a_1 := 1$$
$$a_2 := 2.386294361$$
$$a_3 := 2.149938394$$
$$a_4 := 2.146194257$$
$$a_5 := 2.146193221$$
$$a_6 := 2.146193221$$
$$a_7 := 2.146193221$$
$$a_8 := 2.146193221 \quad \textbf{(6)}$$

> **for** i **from** 2 **to** 8 **do** $print(i, 2 \cdot f(a[i]))$; **end do**;

2, 0.333116292
3, 0.005110998
4, 0.000001414
5, 0.

6, 0.
7, 0.
8, 0. **(7)**

```
>
```

Hemos usado arriba que la cota inferior de la derivada es $1/2$ por eso dividimos por 1/2.

Ejercicio 3-ii)

Ejemplo de estimación de la tolerancia al error en el método de Newton.

Se trata de controlar:
$$
|x_n-x_{n-1}| \le TOL,
$$
donde "TOL" es la tolerancia.

Para introducir las derivadas de f como funciones que evalúan numéricamente pueden usarse los comandos $D(f)$ y $D[1,1](f)$ para la primera y para la segunda derivada.

Ver al final para la derivada segunda.

Debajo, el interval es [a1,b1].

Las iteradas del método de Newton se designan por a[i]; donde a[1] es el dato inicial.

```
> unassign('f');
> f := x→ln(x-1) + cos(x - 1);
```

$$f := x \to \ln(x-1) + \cos(x-1) \quad \textbf{(1)}$$

```
>
> a1 := 1.3; b1 := 1.6;
```

$$a1 := 1.3$$
$$b1 := 1.6 \quad \textbf{(2)}$$

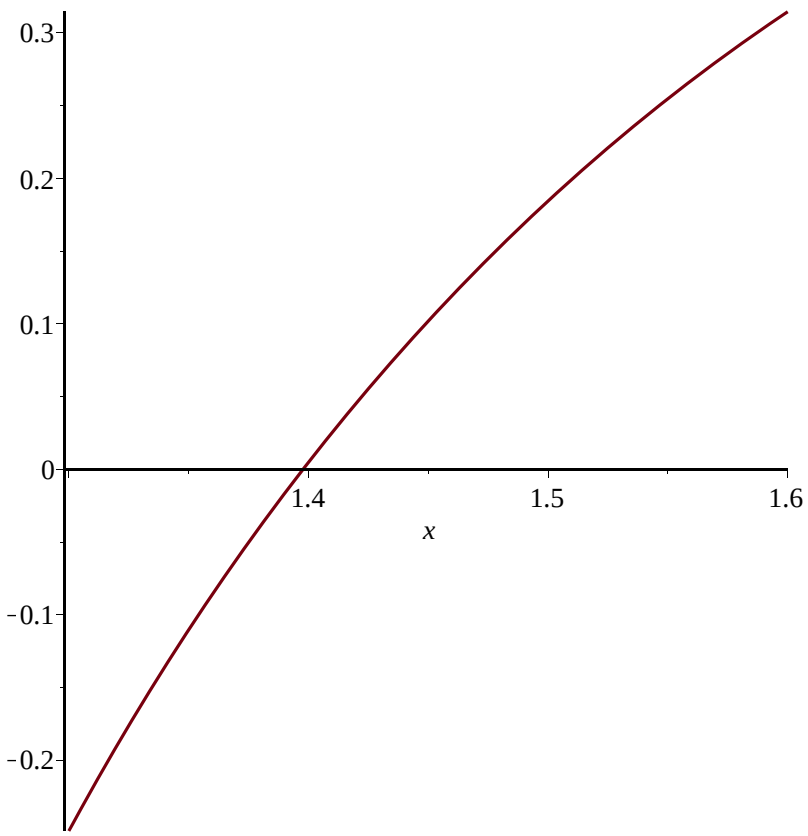
```
> unassign('g');
> g := D(f);
```

$$g := x \to \frac{1}{x-1} - \sin(x-1) \quad \textbf{(3)}$$

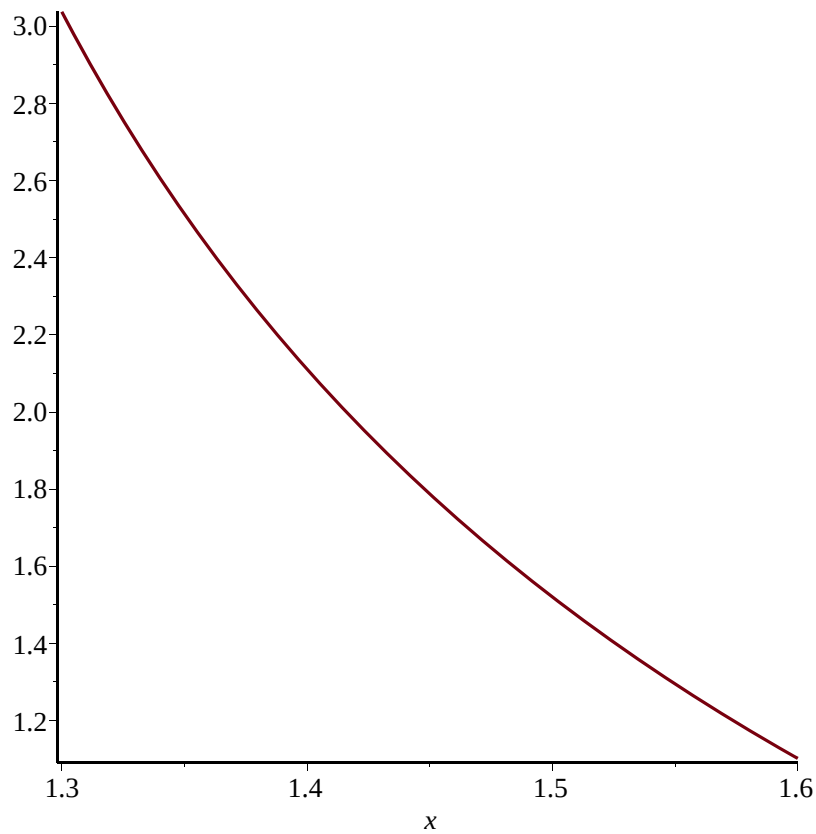
```
>
> evalf(f(a1)/g(a1)); evalf(f(b1)/g(b1));
```

$$-0.08184713957$$
$$0.2853930003 \quad \textbf{(4)}$$

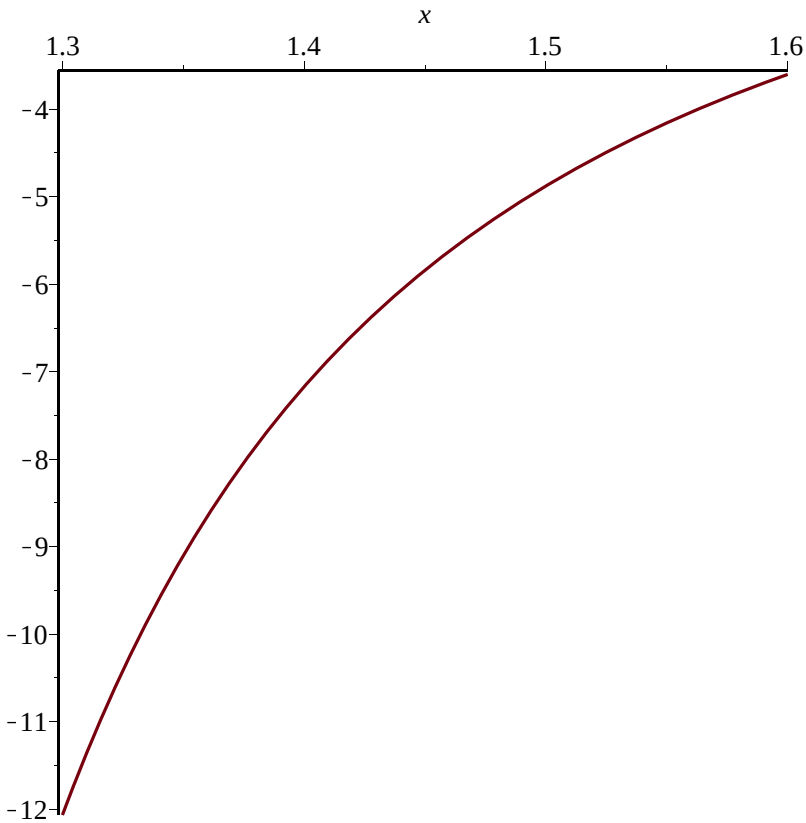
```
> unassign('h');
> plot(f(x), x = a1 ..b1);
```



> $plot(diff(f(x), x), x = a1 .. b1);$




```
> plot(diff(f(x), x$2), x = a1 ..b1);
```



> $unassign('a'); unassign('b');$

> $a[1] := 1.3;$ **for** i **from** 2 **to** 8 **do** $a[i] := evalf\left(a[i-1] - \frac{f(a[i-1])}{g(a[i-1])}\right);$ **end do**;

$$a_1 := 1.3$$
$$a_2 := 1.381847140$$
$$a_3 := 1.397320733$$
$$a_4 := 1.397748164$$
$$a_5 := 1.397748476$$
$$a_6 := 1.397748476$$
$$a_7 := 1.397748476$$
$$a_8 := 1.397748476 \quad \textbf{(5)}$$

> **for** i **from** 2 **to** 8 **do** $print(i, a[i] - a[i-1])$; **end do**;

2, 0.081847140

3, 0.015473593

4, 0.000427431

5, $3.12\ 10^{-7}$

6, 0.
7, 0.
8, 0. **(6)**

```
>
```

Ejercicio 3-iv), en el intervalo [0,1]

Ejemplo de estimación de la tolerancia al error en el método de Newton.

Se trata de controlar:
$$
|x_n-x_{n-1}| \le TOL,
$$
donde "TOL" es la tolerancia.

Para introducir las derivadas de f como funciones que evalúan numéricamente pueden usarse los comandos $D(f)$ y $D[1,1](f)$ para la primera y para la segunda derivada.

Ver al final para la derivada segunda.

Debajo, el interval es [a1,b1].

Las iteradas del método de Newton se designan por a[i]; donde a[1] es el dato inicial.

```
> unassign('f');
> f := x→exp(x) − 3·x^2;
```

$$f := x \rightarrow e^x - 3\,x^2 \quad (1)$$

```
>
> a1 := 0.5; b1 := 1;
```

$$a1 := 0.5$$
$$b1 := 1 \quad (2)$$

```
> unassign('g');
> g := D(f);
```

$$g := x \rightarrow e^x - 6\,x \quad (3)$$

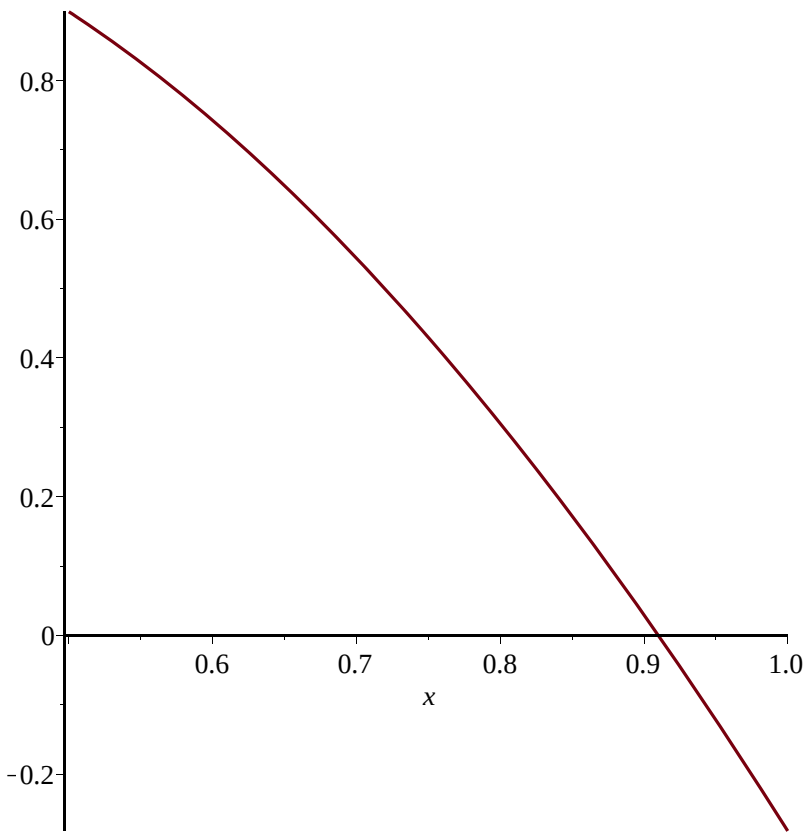
```
> g(x);
```

$$e^x - 6\,x \quad (4)$$

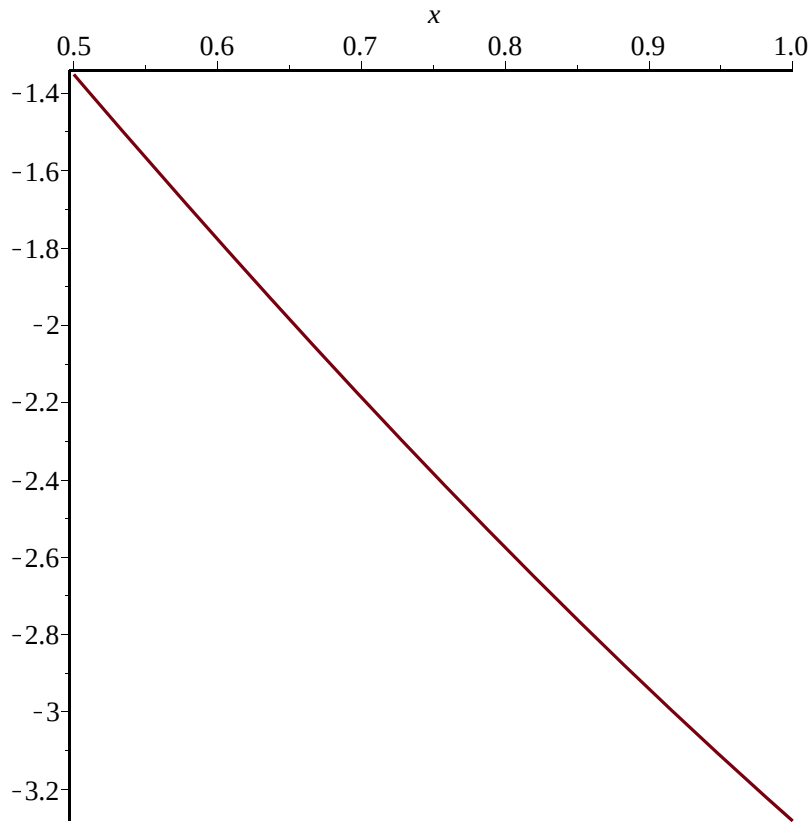
```
> evalf(f(a1)/g(a1)); evalf(f(b1)/g(b1));
```

$$-0.6650894828$$
$$0.08584471830 \quad (5)$$

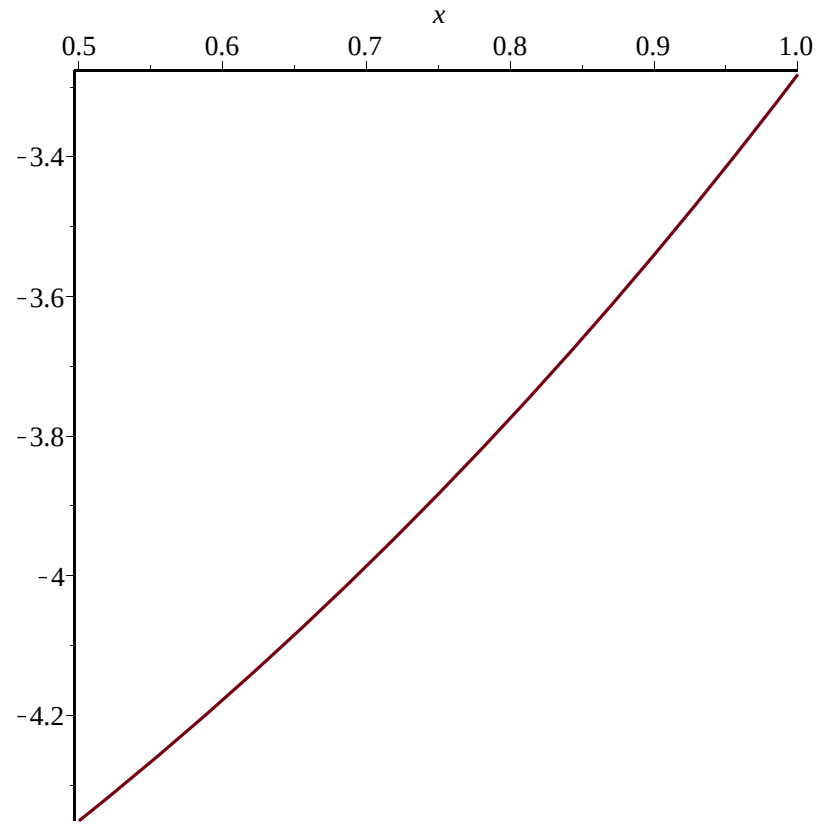
```
> unassign('h');
> plot(f(x), x = a1 ..b1);
```



> $plot(diff(f(x), x), x = a1..b1);$



> $plot(diff(f(x), x\$2), x = a1..b1);$



> $unassign('a'); unassign('b');$

> $a[1] := 1;$ **for** i **from** 2 **to** 8 **do** $a[i] := evalf\left(a[i-1] - \frac{f(a[i-1])}{g(a[i-1])}\right);$ **end do**;

$$a_1 := 1$$
$$a_2 := 0.9141552817$$
$$a_3 := 0.9100176659$$
$$a_4 := 0.9100075726$$
$$a_5 := 0.9100075723$$
$$a_6 := 0.9100075723$$
$$a_7 := 0.9100075723$$
$$a_8 := 0.9100075723 \quad \textbf{(6)}$$

> **for** i **from** 2 **to** 8 **do** $print(i, a[i] - a[i-1])$; **end do**;

2, -0.0858447183

3, -0.0041376158

4, -0.0000100933

5, $-3.\ 10^{-10}$

6, 0.
7, 0.
8, 0. **(7)**

```
>
```

Ejercicio 3-iv), en el intervalo [3,5]

Ejemplo de estimación de la tolerancia al error en el método de Newton.

Se trata de controlar:
$$
|x_n-x_{n-1}| \le TOL,
$$
donde "TOL" es la tolerancia.

Para introducir las derivadas de f como funciones que evalúan numéricamente pueden usarse los comandos $D(f)$ y $D[1,1](f)$ para la primera y para la segunda derivada.

Ver al final para la derivada segunda.

Debajo, el interval es [a1,b1].

Las iteradas del método de Newton se designan por a[i]; donde a[1] es el dato inicial.

```
> unassign('f');
> f := x→exp(x) − 3·x^2;
```

$$f := x \rightarrow e^x - 3\,x^2 \quad \textbf{(1)}$$

```
>
> a1 := 3.5; b1 := 5; evalf(exp(3));
```

$$a1 := 3.5$$
$$b1 := 5$$
$$20.08553692 \quad \textbf{(2)}$$

```
> unassign('g');
> g := D(f);
```

$$g := x \rightarrow e^x - 6\,x \quad \textbf{(3)}$$

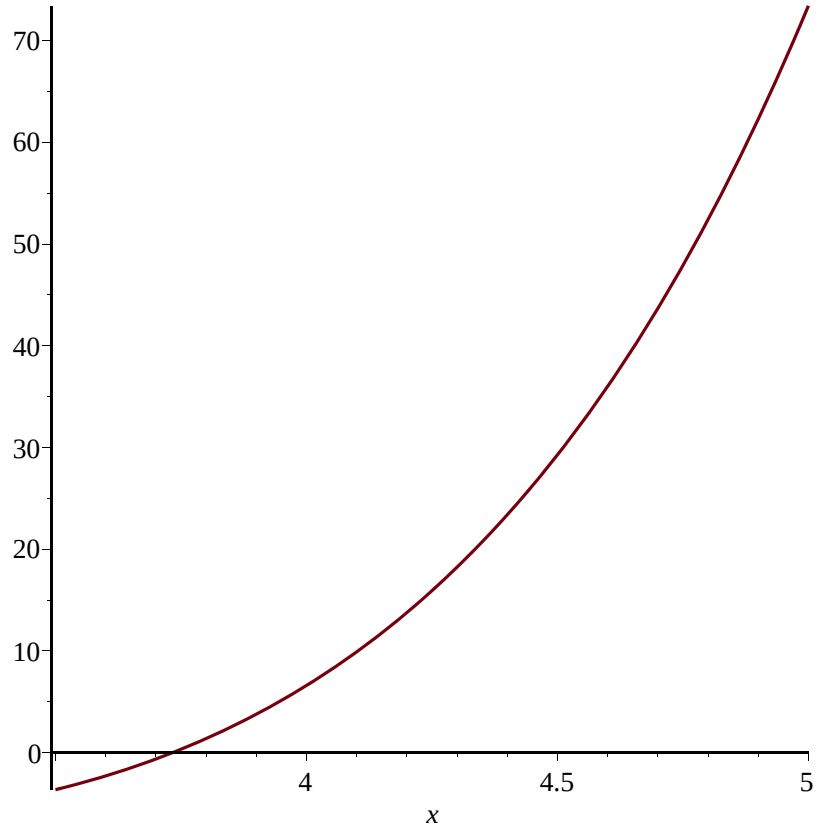
```
> g(x);
```

$$e^x - 6\,x \quad \textbf{(4)}$$

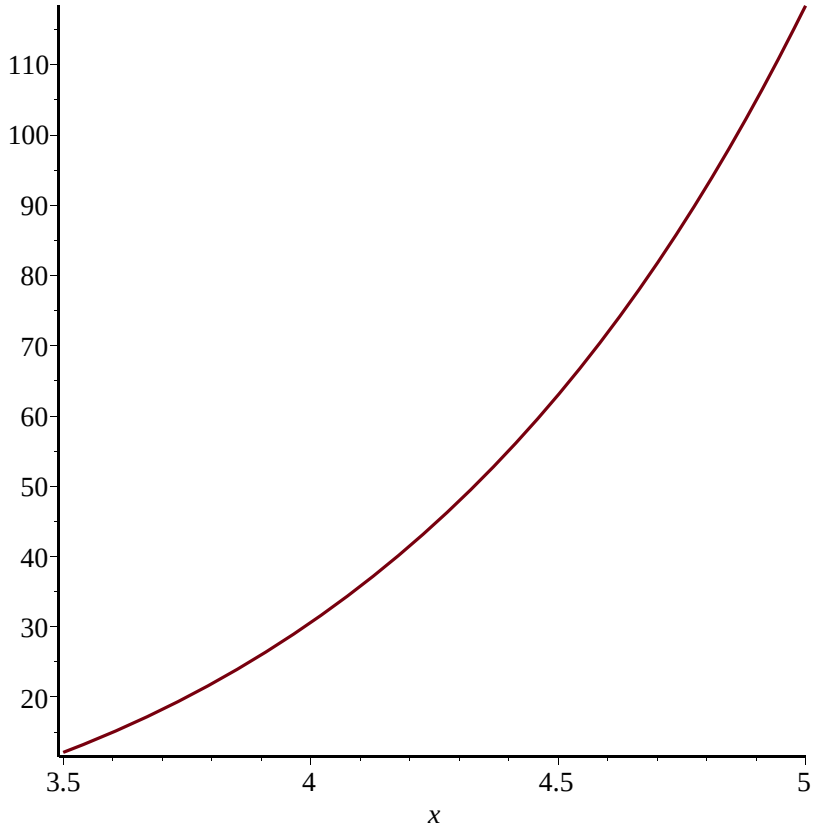
```
> evalf(f(a1)/g(a1)); evalf(f(b1)/g(b1));
```

$$-0.2999927739$$
$$0.6199746688 \quad \textbf{(5)}$$

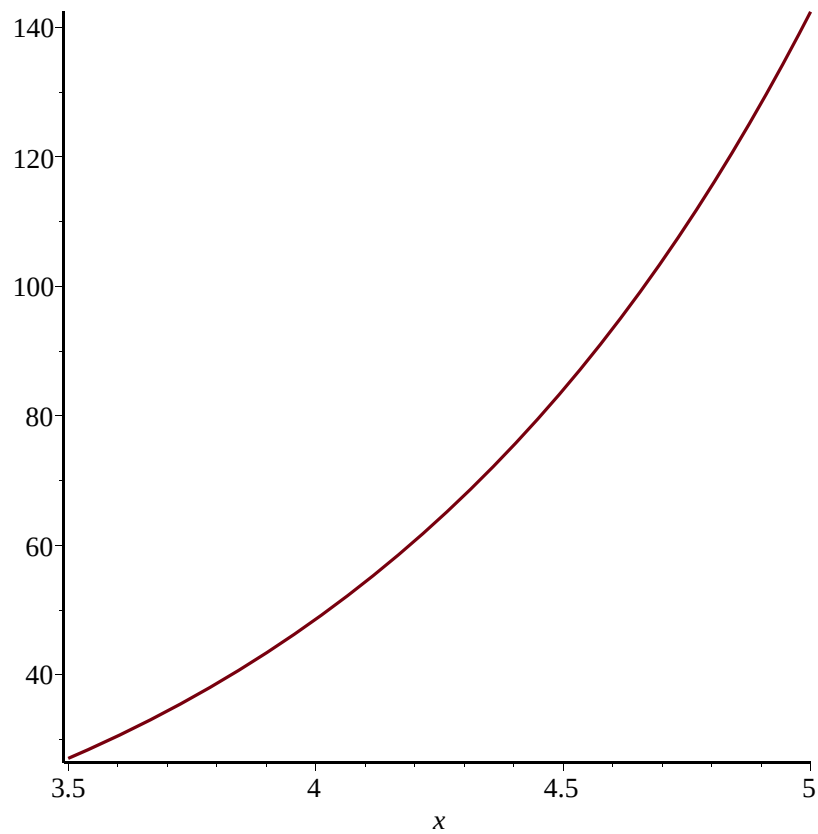
```
> unassign('h');
> plot(f(x), x = a1 ..b1);
```



> $plot(diff(f(x), x), x = a1..b1);$



> $plot(diff(f(x), x\$2), x = a1..b1);$



> $unassign('a'); unassign('b');$

> $a[1] := 5;$ **for** i **from** 2 **to** 8 **do** $a[i] := evalf\left(a[i-1] - \frac{f(a[i-1])}{g(a[i-1])}\right);$ **end do**;

$$a_1 := 5$$
$$a_2 := 4.380025331$$
$$a_3 := 3.963926934$$
$$a_4 := 3.772599183$$
$$a_5 := 3.734461563$$
$$a_6 := 3.733080790$$
$$a_7 := 3.733079028$$
$$a_8 := 3.733079029 \quad \textbf{(6)}$$

> **for** i **from** 2 **to** 8 **do** $print(i, a[i] - a[i-1])$; **end do**;

2, -0.619974669
3, -0.416098397
4, -0.191327751
5, -0.038137620

$$6, -0.001380773$$
$$7, -0.000001762$$
$$8, 1.\ 10^{-9} \tag{7}$$

>

Ejercicio 4

En este ejercicio hay que estimar el error absoluto para dar cifras significativas.

Para introducir las derivadas de f como funciones que evalúan numéricamente pueden usarse los comandos $D(f)$ y $D[1,1](f)$ para la primera y para la segunda derivada.

Ver al final para la derivada segunda.

Debajo, el intervalo es [a1,b1].

Las iteradas del método de Newton se designan por a[i]; donde a[1] es el dato inicial.

Las salidas "print" se podrían dar de forma más elegante. Prefiero no perder el tiempo en eso.

> $unassign('f');$

> $f := x \rightarrow x^4 + 2 \cdot x^3 - 7 \cdot x^2 + 3;$

$$f := x \rightarrow x^4 + 2\,x^3 - 7\,x^2 + 3 \quad \textbf{(1)}$$

>

> $z1 := evalf\left(\frac{(-3 + \mathrm{sqrt}(65))}{4}\right); z2 := evalf\left(\frac{(-6 + \mathrm{sqrt}(36 + 4 \cdot 6 \cdot 7))}{12}\right);$

$$z1 := 1.265564437$$
$$z2 := 0.6902380720 \quad \textbf{(2)}$$

> $a1 := 0.7; b1 := 1;$

$$a1 := 0.7$$
$$b1 := 1 \quad \textbf{(3)}$$

> $g := \mathrm{D}(f);$

$$g := x \rightarrow 4\,x^3 + 6\,x^2 - 14\,x \quad \textbf{(4)}$$

> $g(x);$

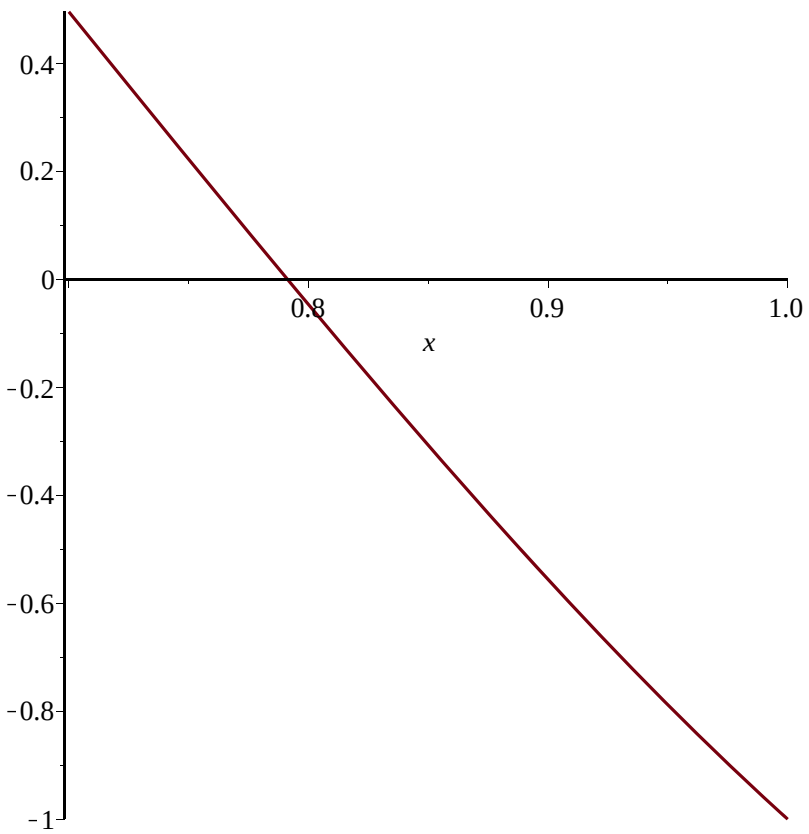
$$e^x - 6\,x \quad \textbf{(5)}$$

> $evalf\left(\frac{f(a1)}{g(a1)}\right); evalf\left(\frac{f(b1)}{g(b1)}\right);$

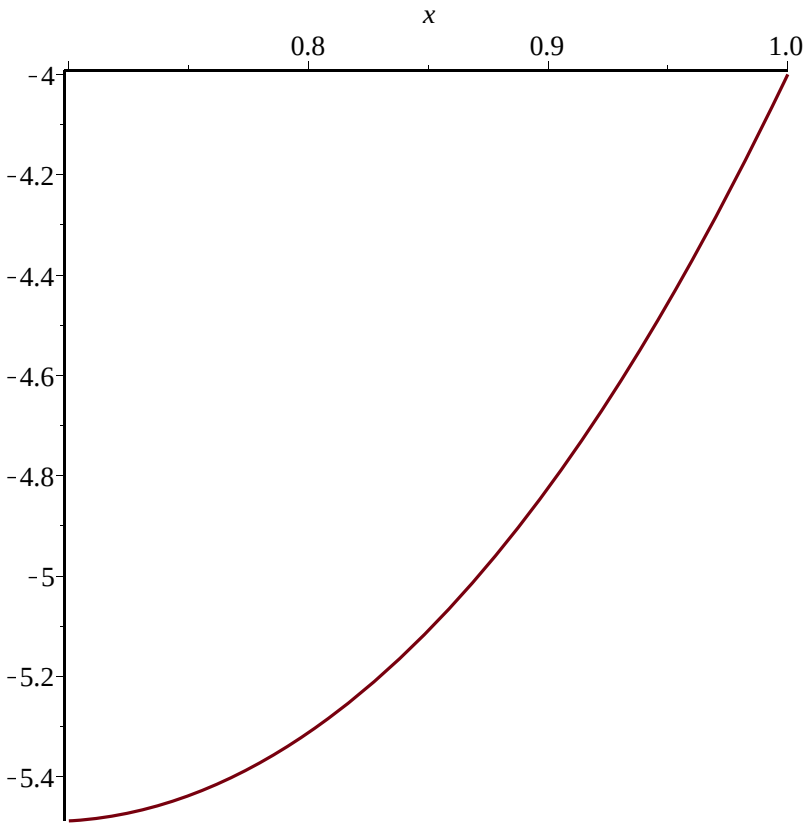
$$-0.09039723032$$
$$0.2500000000 \quad \textbf{(6)}$$

> $unassign('h');$

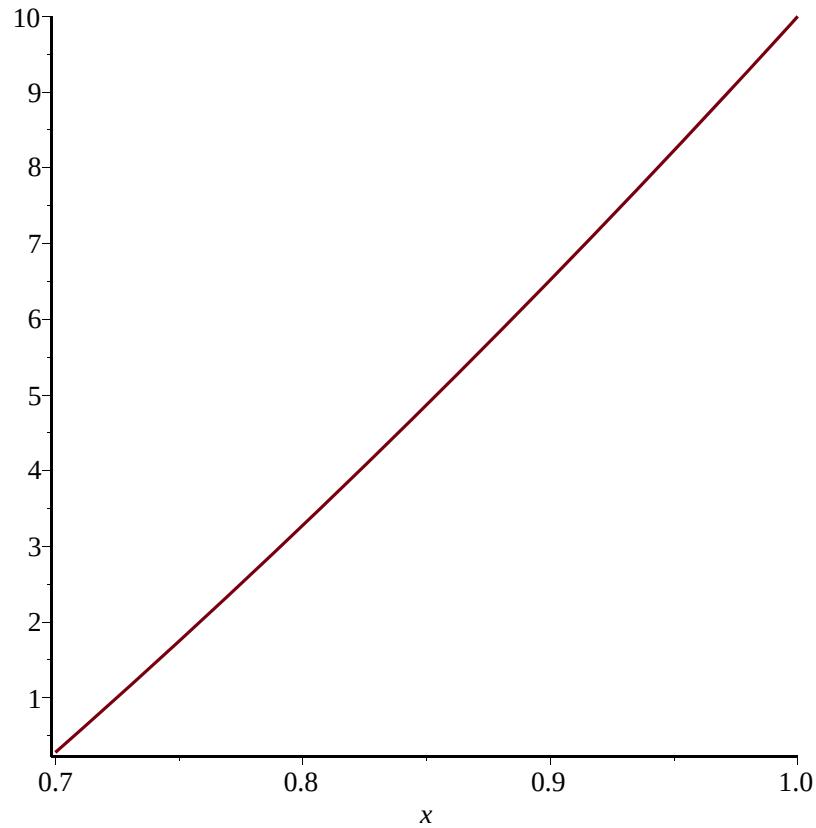
> $plot(f(x), x = a1..b1);$



> $plot(diff(f(x), x), x = a1 .. b1);$




```
> plot(diff(f(x), x$2), x = a1 ..b1);
```



> $unassign('a'); unassign('b');$

> $g(0.7); g(1);$

$$-5.488$$
$$-4 \tag{7}$$

> $a[1] := 1;$ **for** i **from** 2 **to** 8 **do** $a[i] := evalf\left(a[i-1] - \frac{f(a[i-1])}{g(a[i-1])}\right);$ **end do**;

$$a_1 := 1$$
$$a_2 := 0.7500000000$$
$$a_3 := 0.7909482759$$
$$a_4 := 0.7912878151$$
$$a_5 := 0.7912878475$$
$$a_6 := 0.7912878475$$
$$a_7 := 0.7912878475$$
$$a_8 := 0.7912878475 \tag{8}$$

> **for** i **from** 2 **to** 5 **do** $print\left(i, \frac{f(a[i])}{4}\right);$ **end do**;

2, 0.4150390625
3, 0.06144977750
4, 0.002401407500
5, 0.000004225000000 **(9)**

>

Ahora calculamos la segunda raíz de la ecuación. Trabajamos en x mayor que el cero de la derivada
(aproximadamente 1.26).

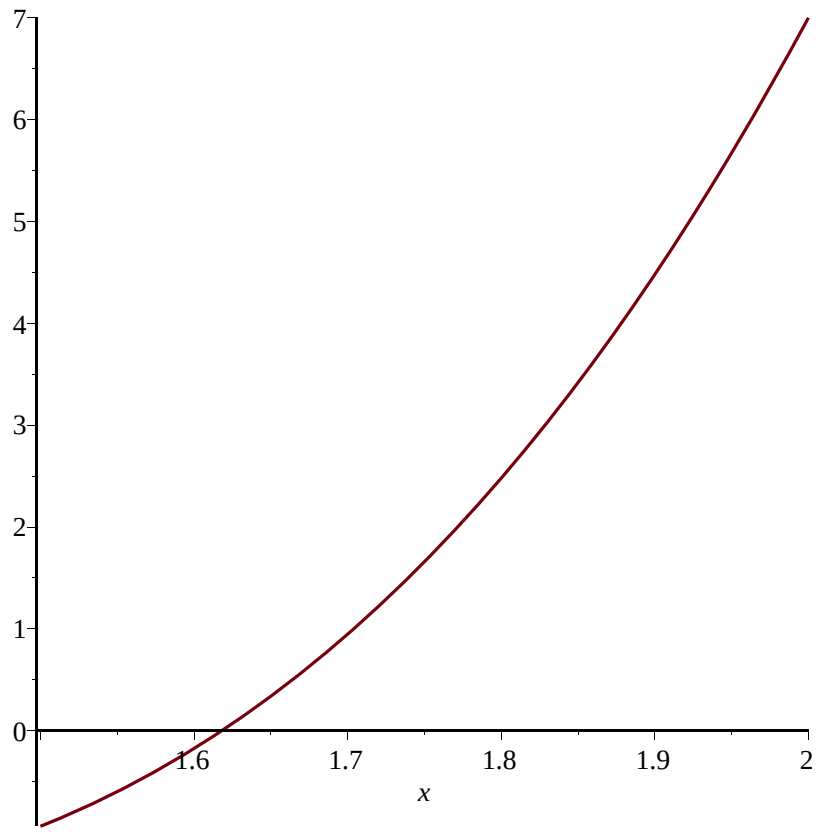
> $a1 := 1.5; b1 := 2;$

$$a1 := 1.5$$
$$b1 := 2 \quad \textbf{(10)}$$

> $evalf\left(\frac{f(a1)}{g(a1)}\right); evalf\left(\frac{f(b1)}{g(b1)}\right);$

$$-0.1562500000$$
$$0.2500000000 \quad \textbf{(11)}$$

> $plot(f(x), x = a1..b1);$



> $a[1] := 2;$ **for** i **from** 2 **to** 5 **do** $a[i] := evalf\left(a[i-1] - \frac{f(a[i-1])}{g(a[i-1])}\right);$ **end do**;

$$a_1 := 2$$
$$a_2 := 1.750000000$$
$$a_3 := 1.641581633$$
$$a_4 := 1.618992858$$
$$a_5 := 1.618035678 \quad \textbf{(12)}$$

> $g(1.5);$

$$6.000 \quad \textbf{(13)}$$

> **for** i **from** 2 **to** 5 **do** $print\left(i, \frac{f(a[i])}{6}\right);$ **end do**;

2, 0.2766927083
3, 0.04096651833
4, 0.001600938333
5, 0.000002816666667 **(14)**

>

Ejercicio 5

Ejemplo de estimación de la tolerancia al error en el método de Newton.

Se trata de controlar:
$$
|x_n-x_{n-1}| \le TOL,
$$
donde "TOL" es la tolerancia.

Para introducir las derivadas de f como funciones que evalúan numéricamente pueden usarse los comandos $D(f)$ y $D[1,1](f)$ para la primera y para la segunda derivada.

Ver al final para la derivada segunda.

Debajo, el interval es [a1,b1].

Las iteradas del método de Newton se designan por a[i]; donde a[1] es el dato inicial.

```
> unassign('f');
> f := x→ln(x^2 + 1) - exp(0.4·x)cos(Pi·x);
```

$$f := x \to \ln\left(x^2+1\right) - e^{0.4\,x}\cos(\pi\,x) \quad (1)$$

```
>
> a1 := -0.5; b1 := -0.4;
```

$$a1 := -0.5$$
$$b1 := -0.4 \quad (2)$$

```
> unassign('g');
> g := D(f);
```

$$g := x \to \frac{2\,x}{x^2+1} - 0.4\,e^{0.4\,x}\cos(\pi\,x) + e^{0.4\,x}\sin(\pi\,x)\,\pi \quad (3)$$

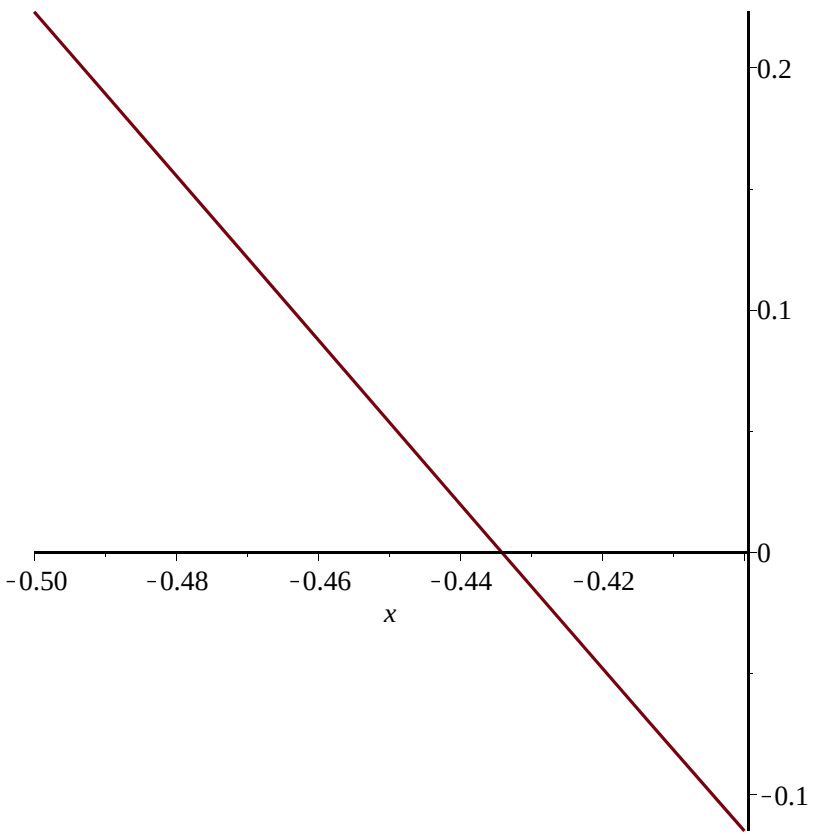
```
> g(x);
```

$$\frac{2\,x}{x^2+1} - 0.4\,e^{0.4\,x}\cos(\pi\,x) + e^{0.4\,x}\sin(\pi\,x)\,\pi \quad (4)$$

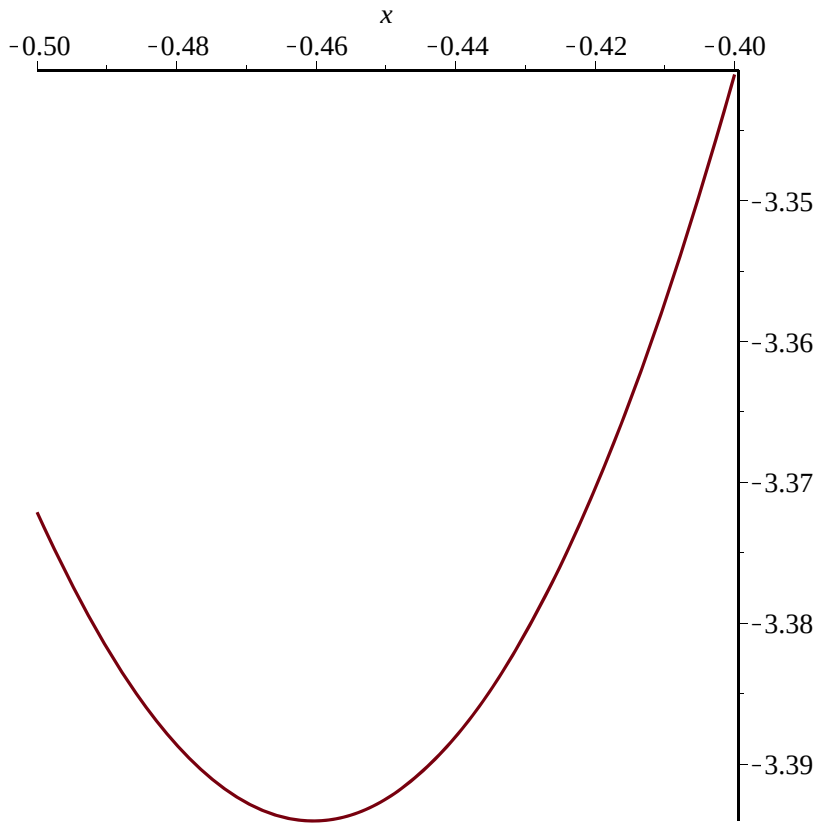
```
> evalf(f(a1)/g(a1)); evalf(f(b1)/g(b1));
```

$$-0.06617310453$$
$$0.03439246836 \quad (5)$$

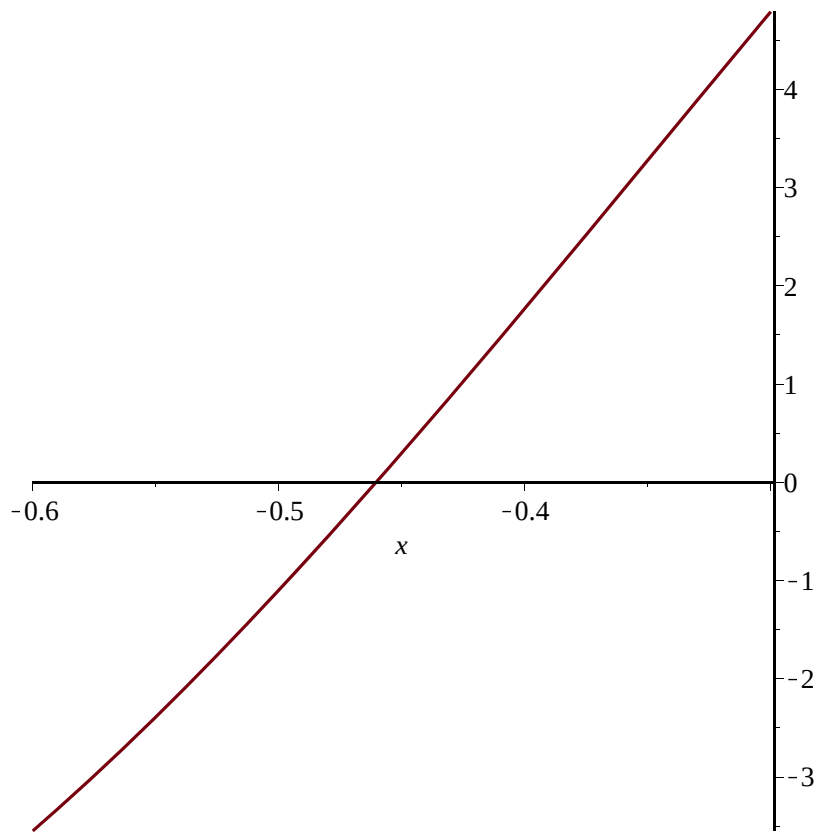
```
> unassign('h');
> plot(f(x), x = a1 ..b1);
```



> $plot(diff(f(x), x), x = a1..b1);$




```
> plot(diff(f(x), x$2), x = a1 ..b1);
```



> $unassign('a'); unassign('b');$

> $a[1] := -0.4;$ **for** i **from** 2 **to** 8 **do** $a[i] := evalf\left(a[i-1] - \frac{f(a[i-1])}{g(a[i-1])}\right);$ **end do**;

$$a_1 := -0.4$$
$$a_2 := -0.4343924684$$
$$a_3 := -0.4341430540$$
$$a_4 := -0.4341430474$$
$$a_5 := -0.4341430473$$
$$a_6 := -0.4341430472$$
$$a_7 := -0.4341430473$$
$$a_8 := -0.4341430472 \quad \textbf{(6)}$$

> **for** i **from** 2 **to** 8 **do** $print(i, a[i] - a[i-1])$; **end do**;

2, -0.0343924684

3, 0.0002494144

4, $6.6\ 10^{-9}$

$$5,\ 1.\ 10^{-10}$$
$$6,\ 1.\ 10^{-10}$$
$$7,\ -1.\ 10^{-10}$$
$$8,\ 1.\ 10^{-10} \tag{7}$$

>

D[1, 1](f) (x);

$$\frac{2}{x^2+1} - \frac{4\,x^2}{\left(x^2+1\right)^2} - 0.16\,e^{0.4\,x}\cos(\pi\,x) + 0.8\,e^{0.4\,x}\sin(\pi\,x)\,\pi + e^{0.4\,x}\cos(\pi\,x)\,\pi^2 \tag{8}$$

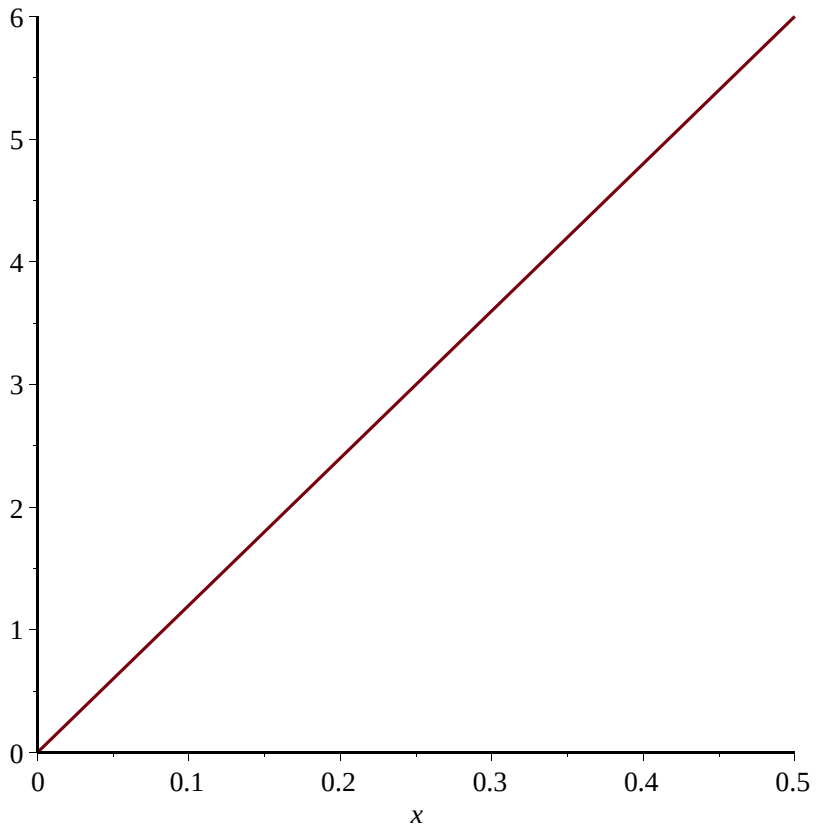
h := D[1, 1](f);

$$x \rightarrow \frac{2}{x^2+1} - \frac{4\,x^2}{\left(x^2+1\right)^2} - 0.16\,e^{0.4\,x}\cos(\pi\,x) + 0.8\,e^{0.4\,x}\sin(\pi\,x)\,\pi + e^{0.4\,x}\cos(\pi\,x)\,\pi^2 \tag{9}$$

$h(5)$;

$$60 \tag{10}$$

$plot(h(x), x = 0\,..0.5)$;


6
5
4
3
2
1
0
0
0.1
0.2
0.3
0.4
0.5
x

Ejercicio 9

En este ejercicio se pide TOL menor que 10^-4. Se busca la mínima distancia entre los puntos de la gráfica de $y=x^2$ y el punto $(1,0)$.

Para introducir las derivadas de f como funciones que evalúan numéricamente pueden usarse los comandos $D(f)$ y $D[1,1](f)$ para la primera y para la segunda derivada.

Ver al final para la derivada segunda.

Debajo, el intervalo es [a1,b1].

Las iteradas del método de Newton se designan por a[i]; donde a[1] es el dato inicial.

Las salidas "print" se podrían dar de forma más elegante. Prefiero no perder el tiempo en eso.

```
> unassign('f');
> f := x→2·x^3 + x − 1;
```

$$f := x \to 2\,x^3 + x - 1 \quad (1)$$

```
>
> a1 := 0.3; b1 := 1;
```

$$a1 := 0.3$$
$$b1 := 1 \quad (2)$$

```
> g := D(f);
```

$$g := x \to 6\,x^2 + 1 \quad (3)$$

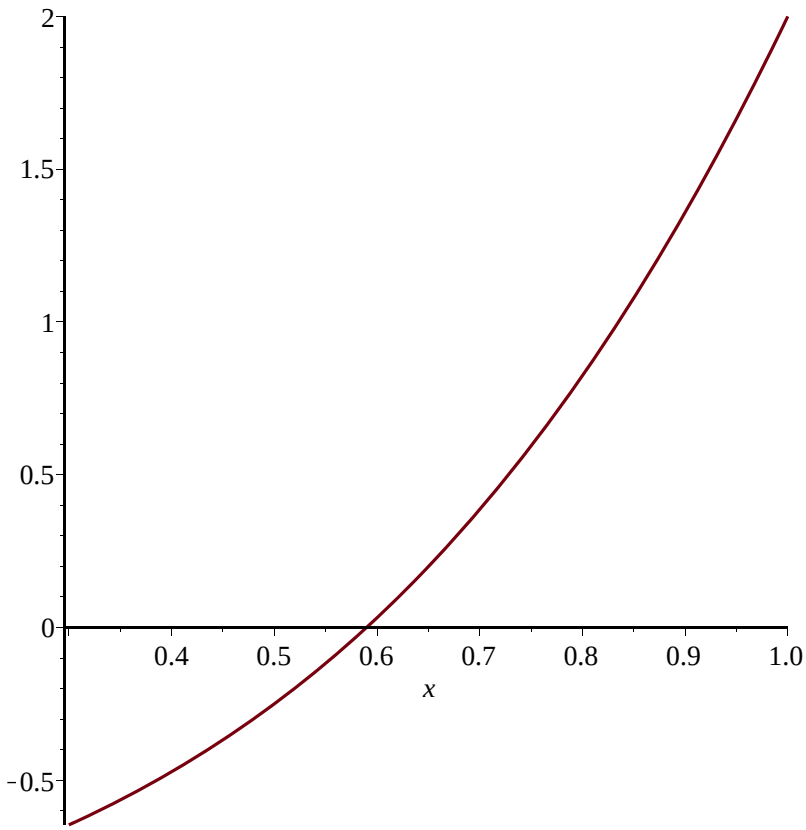
```
> g(x);
```

$$6\,x^2 + 1 \quad (4)$$

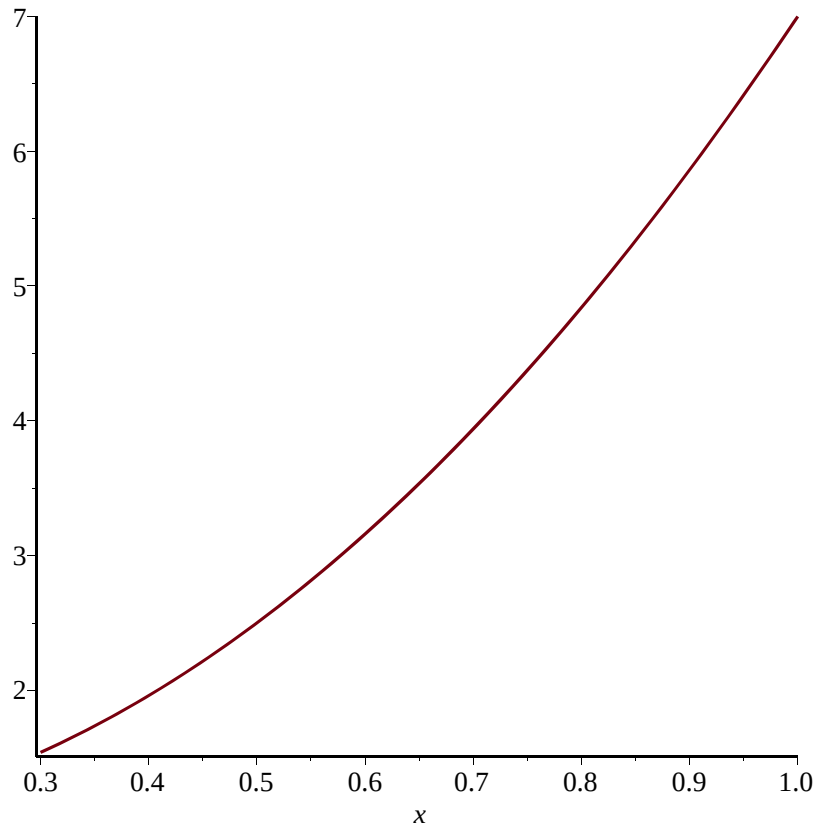
```
> evalf(f(a1)/g(a1)); evalf(f(b1)/g(b1));
```

$$-0.4194805195$$
$$0.2857142857 \quad (5)$$

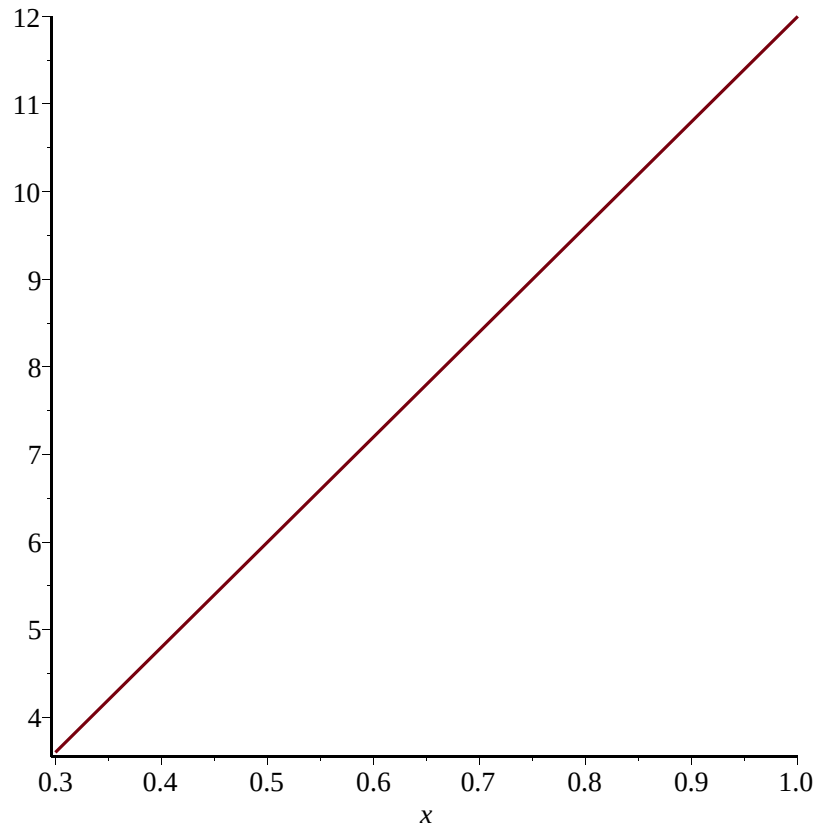
```
> unassign('h');
> plot(f(x), x = a1 ..b1);
```



> $plot(diff(f(x), x), x = a1..b1);$



> $plot(diff(f(x), x\$2), x = a1..b1);$



> $unassign('a'); unassign('b');$

> $g(0.7); g(1);$

$$-5.488$$
$$-4 \tag{6}$$

> $a[1] := 1;$ **for** i **from** 2 **to** 8 **do** $a[i] := evalf\left(a[i-1] - \frac{f(a[i-1])}{g(a[i-1])}\right);$ **end do**;

$$a_1 := 1$$
$$a_2 := 0.7142857143$$
$$a_3 := 0.6051687007$$
$$a_4 := 0.5900220423$$
$$a_5 := 0.5897545942$$
$$a_6 := 0.5897545122$$
$$a_7 := 0.5897545123$$
$$a_8 := 0.5897545123 \tag{7}$$

> **for** i **from** 2 **to** 8 **do** $print(i, a[i] - a[i-1]);$ **end do**;

$$2, -0.2857142857$$

$$3, -0.1091170136$$
$$4, -0.0151466584$$
$$5, -0.0002674481$$
$$6, -8.20\ 10^{-8}$$
$$7, 1.\ 10^{-10}$$
$$8, 0. \tag{8}$$

>

Ejercicio 10

En este ejercicio se pide TOL menor que 10^-4.
Se trata de hallar la mínima distancia entre la
curva $y=1/x$ y el punto $(2,1)$.

Para introducir las derivadas de f como funciones que evalúan numéricamente pueden usarse los comandos $D(f)$ y $D[1,1](f)$ para la primera y para la segunda derivada.

Ver al final para la derivada segunda.

Debajo, el intervalo es [a1,b1].

Las iteradas del método de Newton se designan por a[i]; donde
a[1] es el dato inicial.

Las salidas "print" se podrían dar de forma más elegante. Prefiero no perder el tiempo en eso.

```
> unassign('f');
> f := x→x^4-2·x^3 + x - 1;
```

$$f := x \to x^4 - 2\,x^3 + x - 1 \quad (1)$$

```
>
> a1 := 1.7; b1 := 2;
```

$$a1 := 1.7$$
$$b1 := 2 \quad (2)$$

```
> g := D(f);
```

$$g := x \to 4\,x^3 - 6\,x^2 + 1 \quad (3)$$

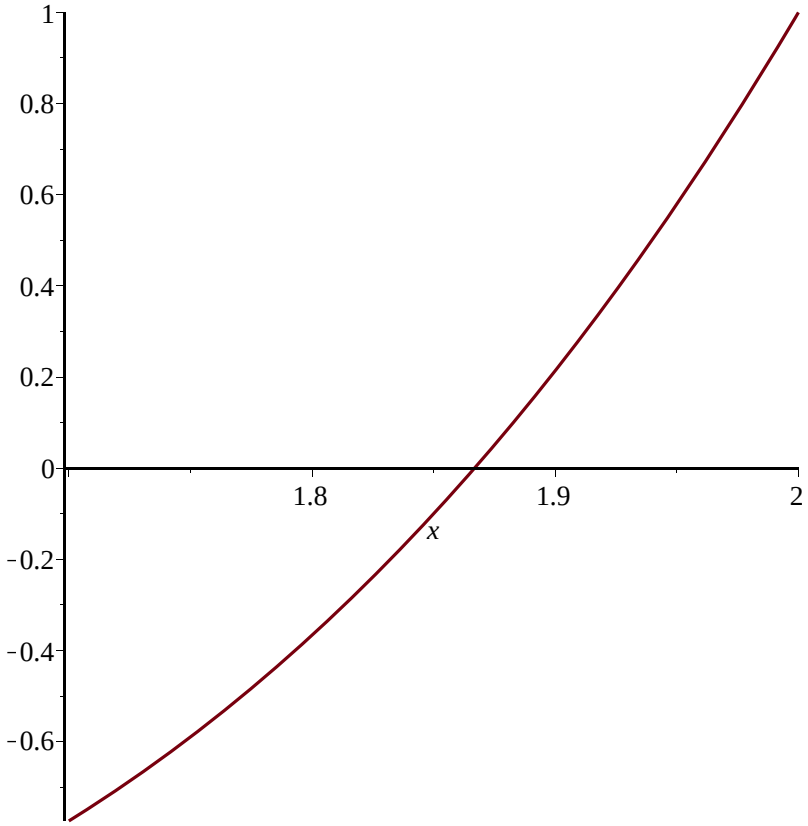
```
> g(1.7);
```

$$3.312 \quad (4)$$

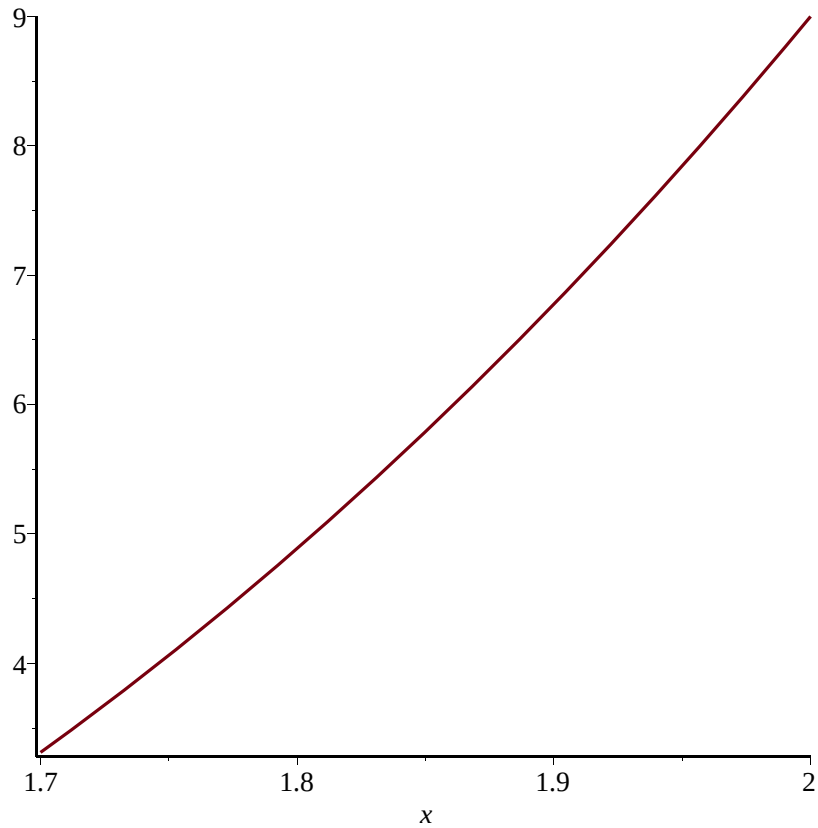
```
> evalf(f(a1)/g(a1)); evalf(f(b1)/g(b1));
```

$$-0.2336654589$$
$$0.1111111111 \quad (5)$$

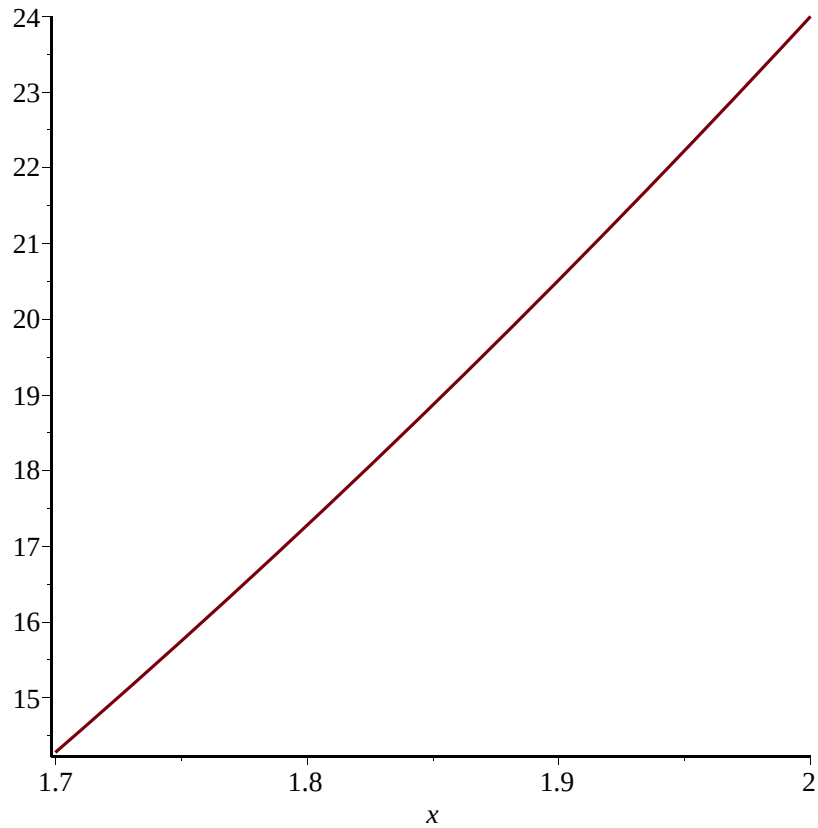
```
> unassign('h');
> plot(f(x), x = a1 ..b1);
```



> $plot(diff(f(x), x), x = a1..b1);$



> $plot(diff(f(x), x\$2), x = a1..b1);$



>

> $a[1] := 2$; **for** i **from** 2 **to** 8 **do** $a[i] := evalf\left(a[i-1] - \frac{f(a[i-1])}{g(a[i-1])}\right)$; **end do**;

$$a_1 := 2$$
$$a_2 := 1.888888889$$
$$a_3 := 1.867504363$$
$$a_4 := 1.866761277$$
$$a_5 := 1.866760399$$
$$a_6 := 1.866760399$$
$$a_7 := 1.866760399$$
$$a_8 := 1.866760399 \quad \textbf{(6)}$$

> **for** i **from** 2 **to** 8 **do** $print(i, a[i] - a[i-1])$; **end do**;

2, -0.111111111

3, -0.021384526

4, -0.000743086

5, $-8.78\ 10^{-7}$

6, 0.
7, 0.
8, 0. **(7)**

```
>
```

Ejercicio 11

En este ejercicio se pide hallar la raíz usando MATLAB.

Dos intervalos [-1,0] y [0,1]. Se pide una TOL del orden de 10^-6.

Para introducir las derivadas de f como funciones que evalúan numéricamente pueden usarse los comandos $D(f)$ y $D[1,1](f)$ para la primera y para la segunda derivada.

Ver al final para la derivada segunda.

Debajo, el intervalo es [a1,b1].

Las iteradas del método de Newton se designan por a[i]; donde a[1] es el dato inicial.

Las salidas "print" se podrían dar de forma más elegante. Prefiero no perder el tiempo en eso.

Conviene empezar por los intervalos señalados y después refinar los valores.

```
> unassign('f');
> f := x→ 230·x^4 + 18·x^3 + 9·x^2 − 221·x − 9;
```

$$f := x \to 230\,x^4 + 18\,x^3 + 9\,x^2 - 221\,x - 9 \quad (1)$$

```
> unassign('g');
> a1 := -0.2; b1 := 0;
```

$$a1 := -0.2$$
$$b1 := 0 \quad (2)$$

```
> f(a1); f(b1);
```

$$35.7840$$
$$-9 \quad (3)$$

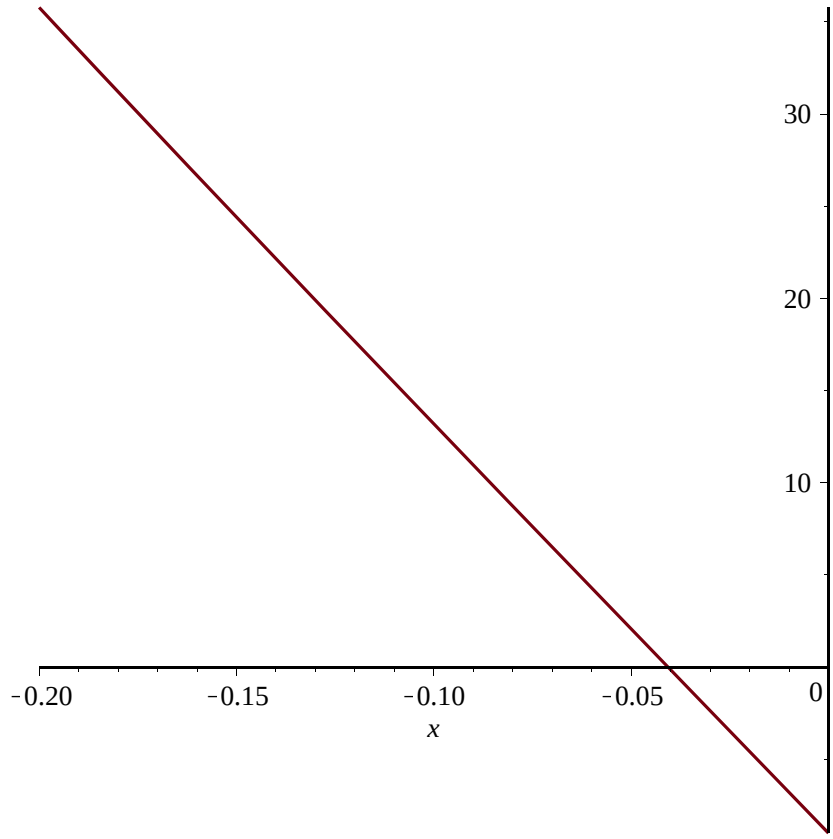
```
> g := D(f);
```

$$g := x \to 920\,x^3 + 54\,x^2 + 18\,x - 221 \quad (4)$$

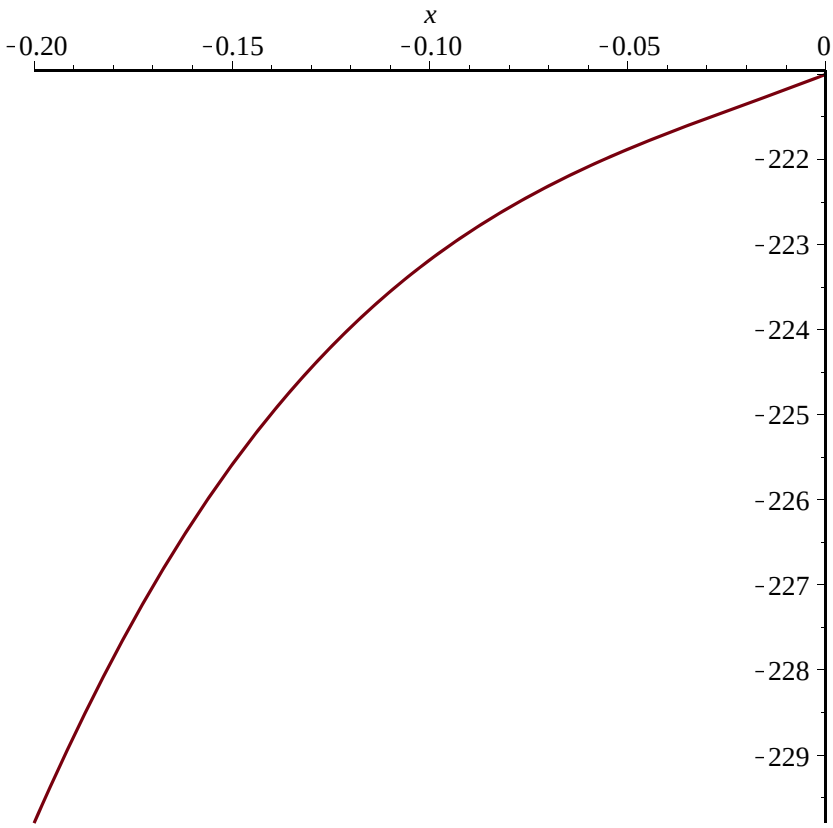
```
>
> evalf(f(a1)/g(a1)); evalf(f(b1)/g(b1));
```

$$-0.1557180157$$
$$0.04072398190 \quad (5)$$

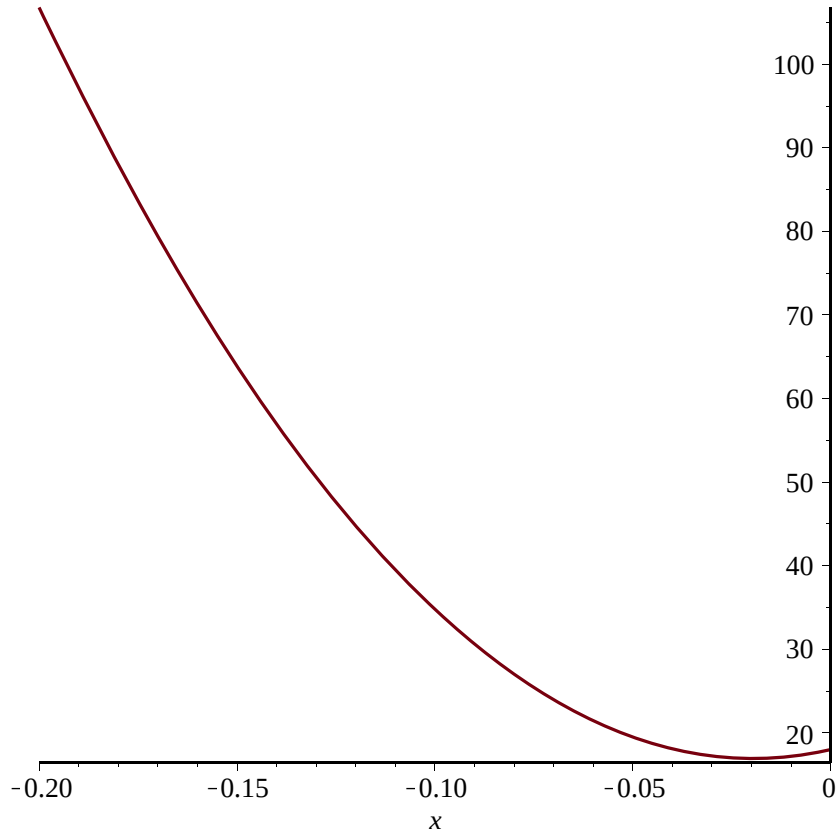
```
> unassign('h');
> plot(f(x), x = a1..b1);
```



> $plot(diff(f(x), x), x = a1..b1);$




```
> plot(diff(f(x), x$2), x = a1 ..b1);
```



>

> $a[1] := a1$; **for** i **from** 2 **to** 8 **do** $a[i] := evalf\left(a[i-1] - \frac{f(a[i-1])}{g(a[i-1])}\right)$; **end do**;

$$a_1 := -0.2$$
$$a_2 := -0.0442819843$$
$$a_3 := -0.04065983485$$
$$a_4 := -0.04065928832$$
$$a_5 := -0.04065928832$$
$$a_6 := -0.04065928832$$
$$a_7 := -0.04065928832$$
$$a_8 := -0.04065928832 \quad \textbf{(6)}$$

> **for** i **from** 2 **to** 8 **do** $print(i, a[i] - a[i-1])$; **end do**;

2, 0.1557180157

3, 0.00362214945

4, $5.4653\ 10^{-7}$

5, 0.

6, 0.
7, 0.
8, 0. **(7)**

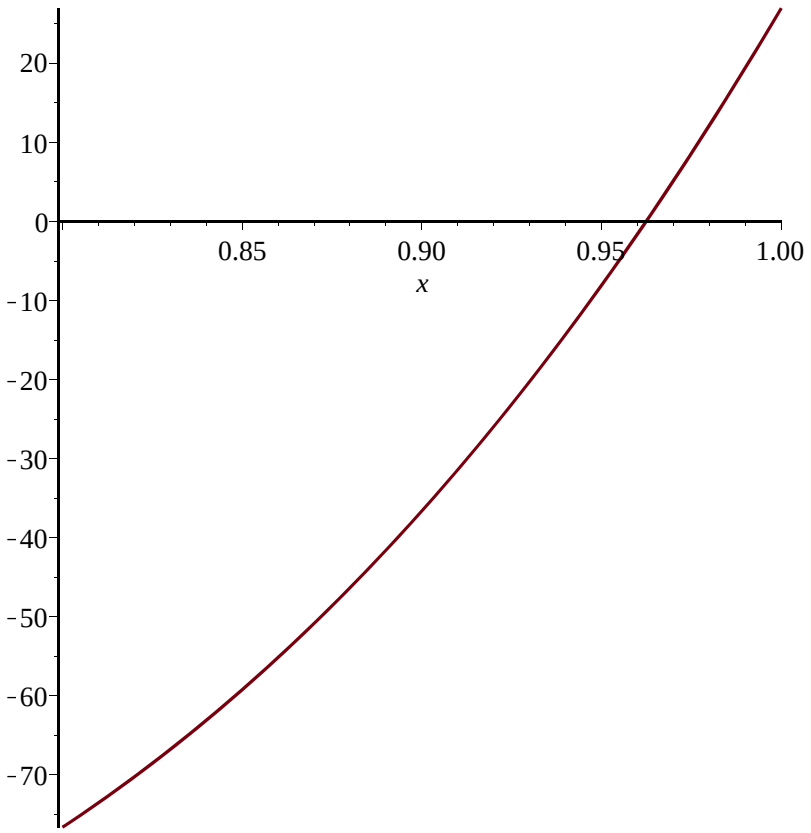
>

Ahora vamos con el intervalo [0,1]

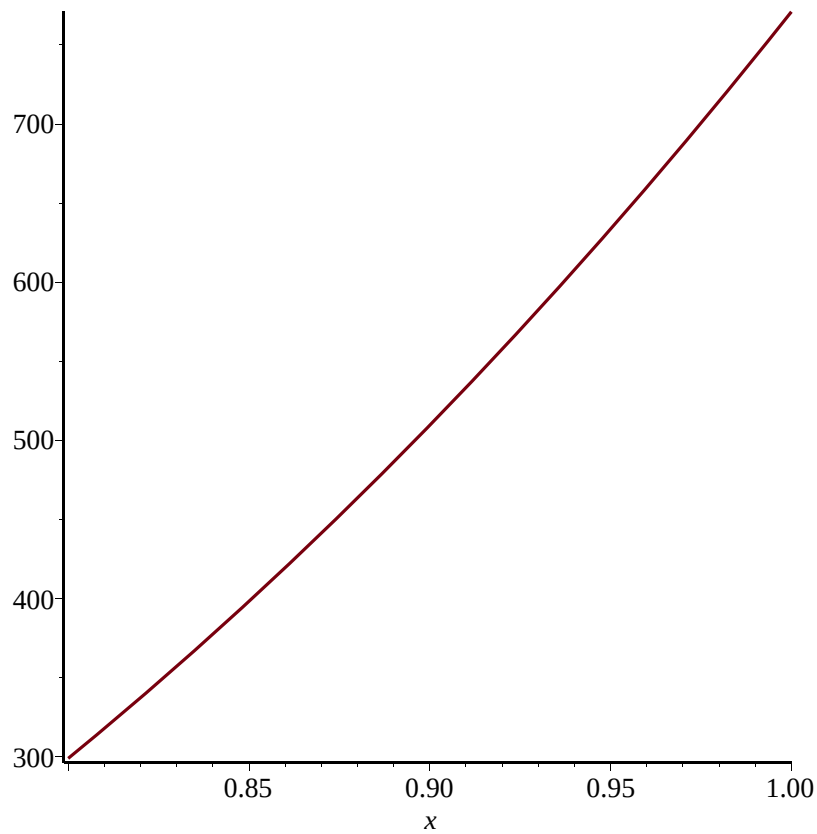
> $a1 := 0.8;\ b1 := 1;$

$a1 := 0.8$
$b1 := 1$ **(8)**

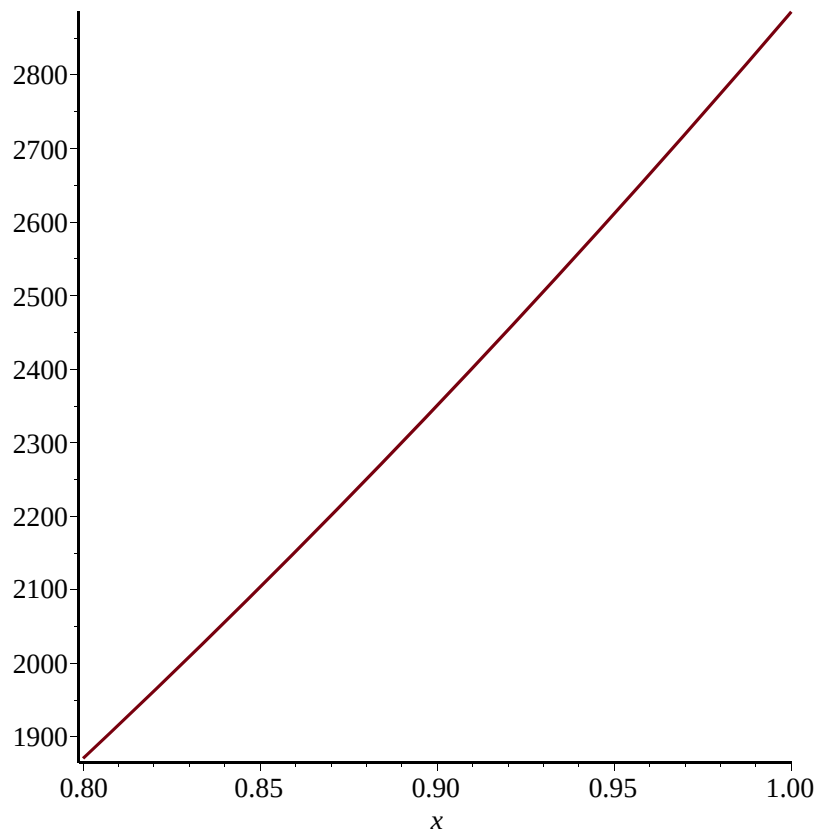
> $plot(f(x), x = a1..b1);$



> $plot(diff(f(x), x), x = a1..b1);$



> $plot(diff(f(x), x\$2), x = a1..b1);$



> $evalf\left(\frac{f(a1)}{g(a1)}\right); evalf\left(\frac{f(b1)}{g(b1)}\right);$

$$-0.2562408027$$
$$0.03501945525 \quad \textbf{(9)}$$

> $a[1] := a1;$ **for** i **from** 2 **to** 8 **do** $a[i] := evalf\left(a[i-1] - \frac{f(a[i-1])}{g(a[i-1])}\right);$ **end do**;

$$a_1 := 0.8$$
$$a_2 := 1.056240803$$
$$a_3 := 0.9765537980$$
$$a_4 := 0.9627864090$$
$$a_5 := 0.9623987210$$
$$a_6 := 0.9623984188$$
$$a_7 := 0.9623984190$$
$$a_8 := 0.9623984187 \quad \textbf{(10)}$$

> **for** i **from** 2 **to** 8 **do** $print(i, a[i] - a[i-1])$; **end do**;

2, 0.256240803

$$3, -0.0796870050$$
$$4, -0.0137673890$$
$$5, -0.0003876880$$
$$6, -3.022\ 10^{-7}$$
$$7, 2.\ 10^{-10}$$
$$8, -3.\ 10^{-10} \tag{11}$$

>

Ejercicio 14

En este ejercicio se pide encontrar un intervalo [1,b] donde aplicar Newton con libertad.

Para introducir las derivadas de f como funciones que evalúan numéricamente pueden usarse los comandos $D(f)$ y $D[1,1](f)$ para la primera y para la segunda derivada.

Ver al final para la derivada segunda.

Debajo, el intervalo es [a1,b1].

Las iteradas del método de Newton se designan por a[i]; donde a[1] es el dato inicial.

Las salidas "print" se podrían dar de forma más elegante. Prefiero no perder el tiempo en eso.

> $unassign('f'); unassign('x');$

> $f := x \rightarrow x^6 - x - 1;$

$$f := x \rightarrow x^6 - x - 1 \quad (1)$$

> $unassign('g');$

> $a1 := 1;\ b1 := \frac{6.0}{5};$

$$a1 := 1$$
$$b1 := 1.200000000 \quad (2)$$

> $f(b1);$

$$0.785984000 \quad (3)$$

> $g := D(f);$

$$g := x \rightarrow 6\,x^5 - 1 \quad (4)$$

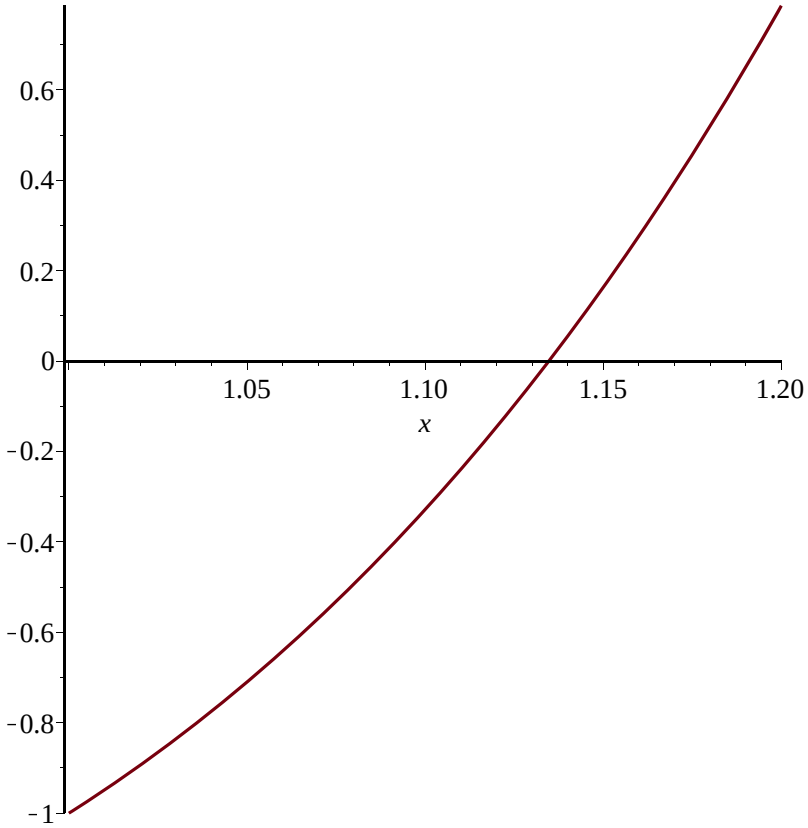
>

> $evalf\left(\frac{f(a1)}{g(a1)}\right); evalf\left(\frac{f(b1)}{g(b1)}\right);$

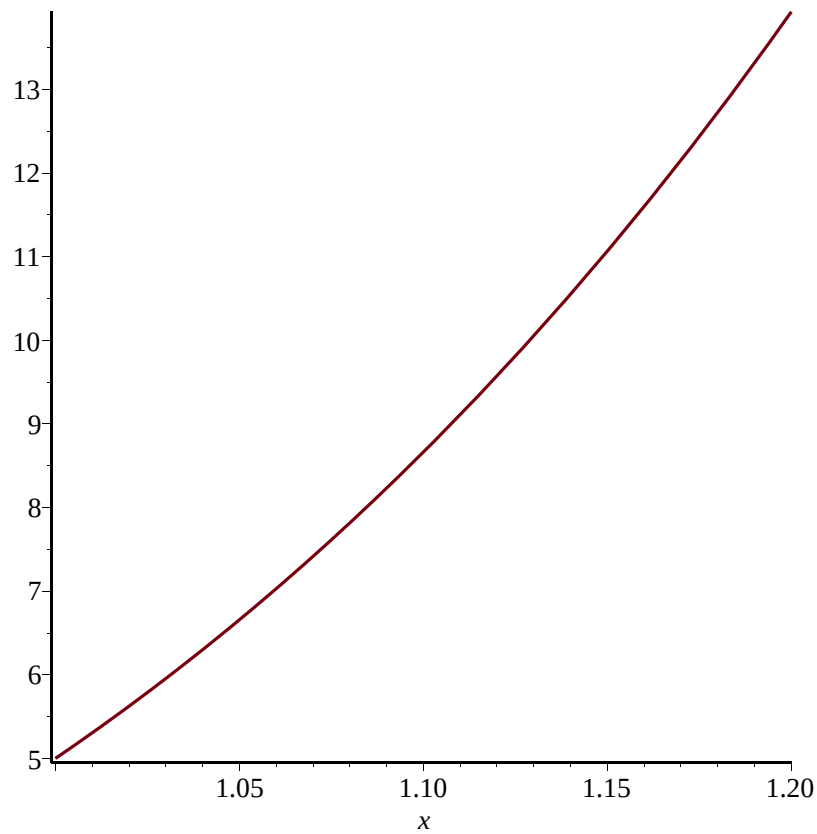
$$-0.2000000000$$
$$0.05642415750 \quad (5)$$

> $unassign('h');$

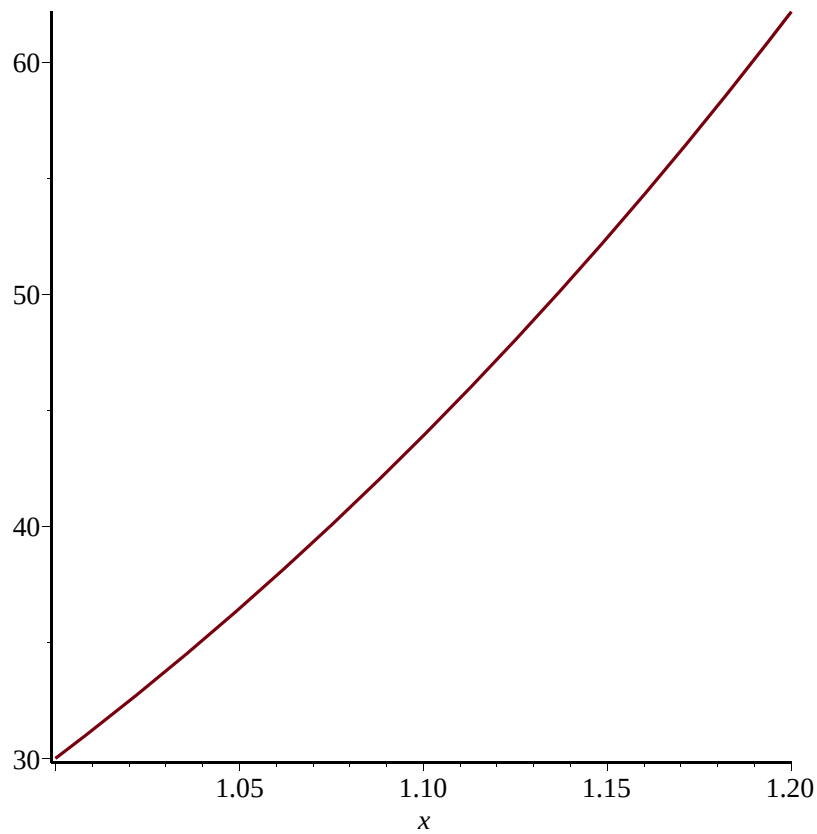
> $plot(f(x), x = a1..b1);$



> $plot(diff(f(x), x), x = a1 .. b1);$




```
> plot(diff(f(x), x$2), x = a1 ..b1);
```



>

> $a[1] := b1$; **for** i **from** 2 **to** 8 **do** $a[i] := evalf\left(a[i-1] - \frac{f(a[i-1])}{g(a[i-1])}\right)$; **end do**;

$$a_1 := 1.200000000$$
$$a_2 := 1.143575842$$
$$a_3 := 1.134909462$$
$$a_4 := 1.134724221$$
$$a_5 := 1.134724138$$
$$a_6 := 1.134724138$$
$$a_7 := 1.134724138$$
$$a_8 := 1.134724138 \quad \textbf{(6)}$$

> **for** i **from** 2 **to** 8 **do** $print(i, a[i] - a[i-1])$; **end do**;

2, -0.056424158

3, -0.008666380

4, -0.000185241

5, $-8.3\ 10^{-8}$

6, 0.
7, 0.
8, 0. **(7)**

```
>
```