Aufgabe: Bestimme die Lösung der Gleichung

```
> x^x=2;
```

$$x^x = 2 \tag{1}$$

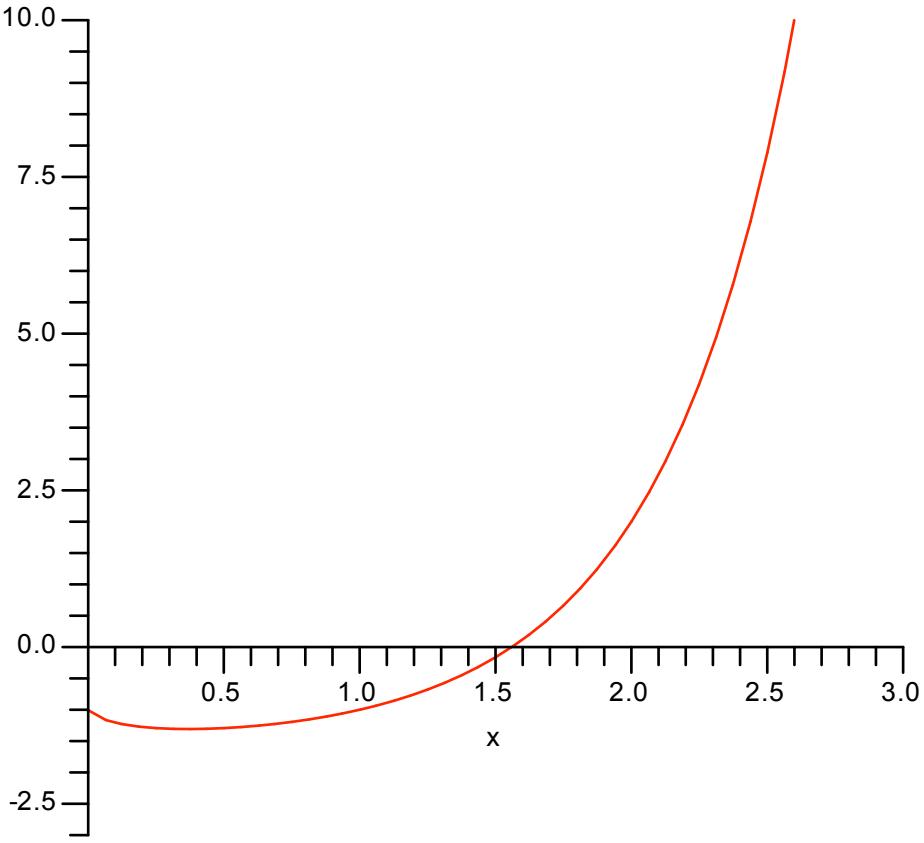
Dafür betrachten wir die Funktion:

```
> f(x) :=x^x-2;
```

$$f(x) := x^x - 2 \tag{2}$$

Hier ist eine Skizze:

```
> plot(x^x-2,x=0.001..3,-3..10);
```



Die ersten beiden Ableitungen

```
> f1(x) := diff(f(x),x);
> f2(x) := diff(f1(x),x);
```

sind offensichtlich positiv für x>1. Darum ist die Funktion streng monoton wachsend und konvex. Insbesondere hat sie genau eine Nullstelle x>1. Wir bestimmen diese Nullstelle näherungsweise mit dem Newton-Verfahren.

$$f1(x) := x^x\,(\ln(x)+1)$$

$$f2(x) := x^x\,(\ln(x)+1)^2 + \frac{x^x}{x} \tag{3}$$

Wir beginnen mit der standardmässig eingestellten Rechengenauigkeit von 10 floating point Dezimalstellen:

```
> Digits := 10;
```

$$Digits := 10 \tag{4}$$

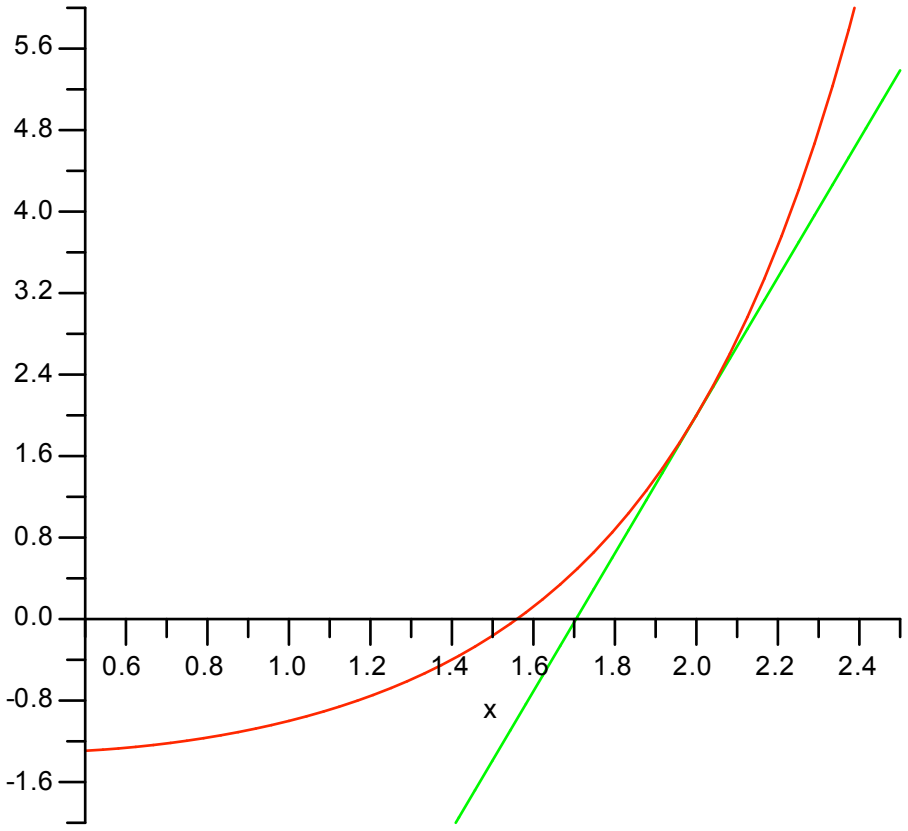
Wir beginnen mit dem Anfangswert

```
> t:=2;
```

$$t := 2 \tag{5}$$

Der jeweils nächste Wert ergibt sich aus der Tangente in dem Punkt (t,f(t)) durch Schneiden mit der x-Achse:

```
> plot([x^x-2,subs(x=t,f(x))+subs(x=t,f1(x))*(x-t)],x=0.5..2.5,-2.
  .6);
```




```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 1.704691946 \tag{6}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 1.577944558 \tag{7}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 1.559924538 \tag{8}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

(9)

$$t := 1.559610563 \quad (9)$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 1.559610470 \quad (10)$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 1.559610469 \quad (11)$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 1.559610469 \quad (12)$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 1.559610469 \quad (13)$$

Offensichtlich hat sich der Wert schon nach 6 Schritten stabilisiert in den ersten 9 Nachkommastellen.

Versuchen wir es noch einmal mit grösserer Rechengenauigkeit:

```
> Digits := 100;
```

$$Digits := 100 \quad (14)$$

```
> t:=2;
```

$$t := 2 \quad (15)$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 1.70469194542517937512809654533837422144167347556305996627986039903908\backslash 75331227713623727823880290755 81 \quad (16)$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 1.57794455747627044569220823373823215106864236086675276070613595024266\backslash 17305423720448511551687743703 67 \quad (17)$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 1.55992453751707899242823044435245682392182514852486060058220214991352\backslash 55611570156563085767305522095 49 \quad (18)$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 1.55961056257717667811411825090521591061446411465339074078530052555994\backslash 47408044122845127734655445198 61 \quad (19)$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 1.55961046946237753625543253382371872673415175680679883477515124483111\backslash 03733103409961649177442336438 03 \quad (20)$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := \quad (21)$$

1.559610469462369349970388768828276676075436727788748868314071926822954299412641408046093730399283014

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$t :=$ (22)

1.559610469462369349970388768765002993284883511843091424723374602608864936778072034298057463948346187

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$t :=$ (23)

1.559610469462369349970388768765002993284883511843091424719594569413973034549590587105413444691283974

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$t :=$ (24)

1.559610469462369349970388768765002993284883511843091424719594569413973034549590587105413444691283974

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$t :=$ (25)

1.559610469462369349970388768765002993284883511843091424719594569413973034549590587105413444691283974

Hier haben bereits 8 Schritte genügt, um den Wert auf 99 Nachkommastellen genau zu berechnen! Dies illustriert die Stärke des Verfahrens.

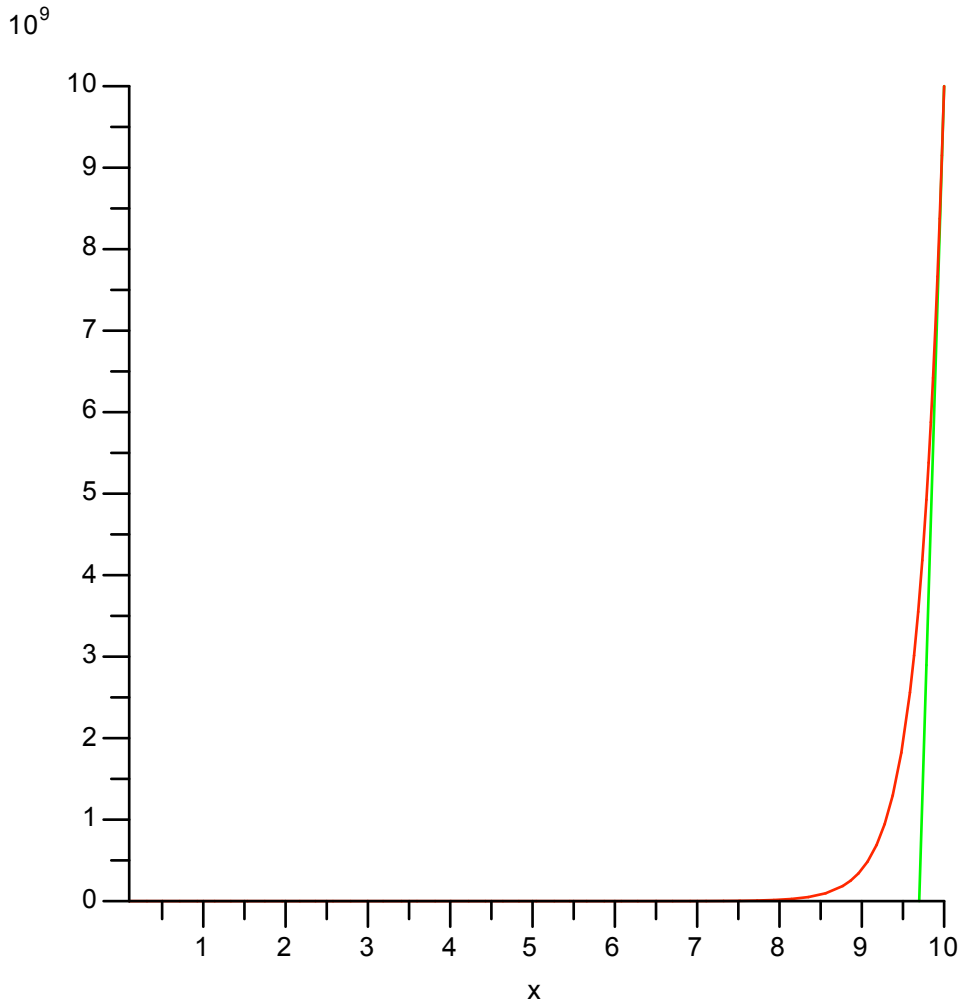
Bei einem ungünstigeren Startwert

```
> t:=10;
```

$t := 10$ (26)

geschieht zum Beispiel folgendes. Da die Tangente fast vertikal ist, ändert sich der Wert von t in einem Schritt relativ wenig:

```
> plot([x^x-2,subs(x=t,f(x))+subs(x=t,f1(x))*(x-t)],x=0.1..10,-10.
  .10000000000);
```




```
> Digits := 10;
```

$$Digits := 10 \tag{27}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 9.697206894 \tag{28}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 9.391568280 \tag{29}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 9.082908423 \tag{30}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 8.771031637 \tag{31}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 8.455718896 \tag{32}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 8.136723664 \tag{33}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 7.813766706 \tag{34}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```
$$t := 7.486529559 \tag{35}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```
$$t := 7.154646231 \tag{36}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```
$$t := 6.817692502 \tag{37}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```
$$t := 6.475171989 \tag{38}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```
$$t := 6.126497830 \tag{39}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```
$$t := 5.770968550 \tag{40}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```
$$t := 5.407736710 \tag{41}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```
$$t := 5.035770310 \tag{42}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```
$$t := 4.653812845 \tag{43}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```
$$t := 4.260367992 \tag{44}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```
$$t := 3.853796758 \tag{45}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```
$$t := 3.432795886 \tag{46}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```
$$t := 2.998023758 \tag{47}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```
$$t := 2.556823225 \tag{48}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```
$$t := 2.134590212 \tag{49}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```
$$t := 1.791270495 \tag{50}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```
$$t := 1.604251568 \tag{51}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```
$$t := 1.561444976 \tag{52}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```
$$t := 1.559613644 \tag{53}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```
$$t := 1.559610469 \tag{54}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 1.559610469 \tag{55}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 1.559610469 \tag{56}$$

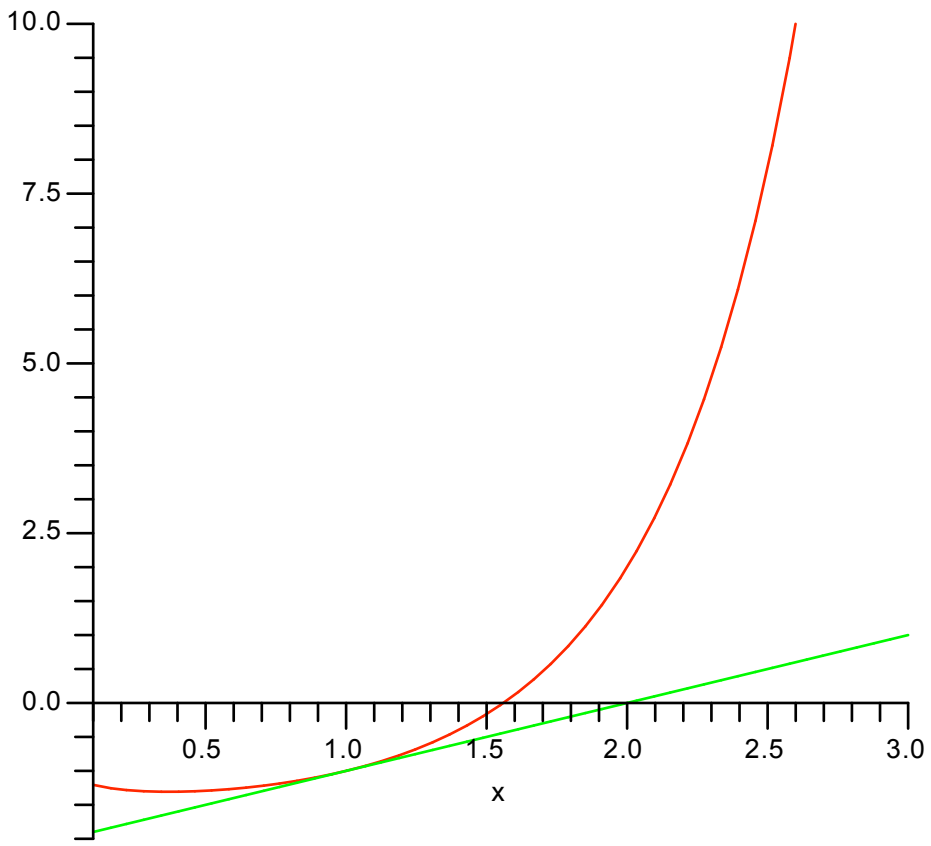
Hier haben wir 27 Schritte gebraucht, um 9 Nachkommastellen genau zu berechnen. Ab dann konvergiert das Verfahren natürlich wieder rasant, wie oben.

Ein zu kleiner Startwert wird zuerst nach rechts geworfen und konvergiert danach von rechts gegen die gesuchte Nullstelle:

```
> t:=1;
```

$$t := 1 \tag{57}$$

```
> plot([x^x-2,subs(x=t,f(x))+subs(x=t,f1(x))*(x-t)],x=0.1..3,-2..10)
  ;
```




```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 2. \tag{58}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 1.704691946 \tag{59}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

(60)

$$t := 1.577944558 \quad (60)$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 1.559924538 \quad (61)$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 1.559610563 \quad (62)$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 1.559610470 \quad (63)$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 1.559610469 \quad (64)$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 1.559610469 \quad (65)$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

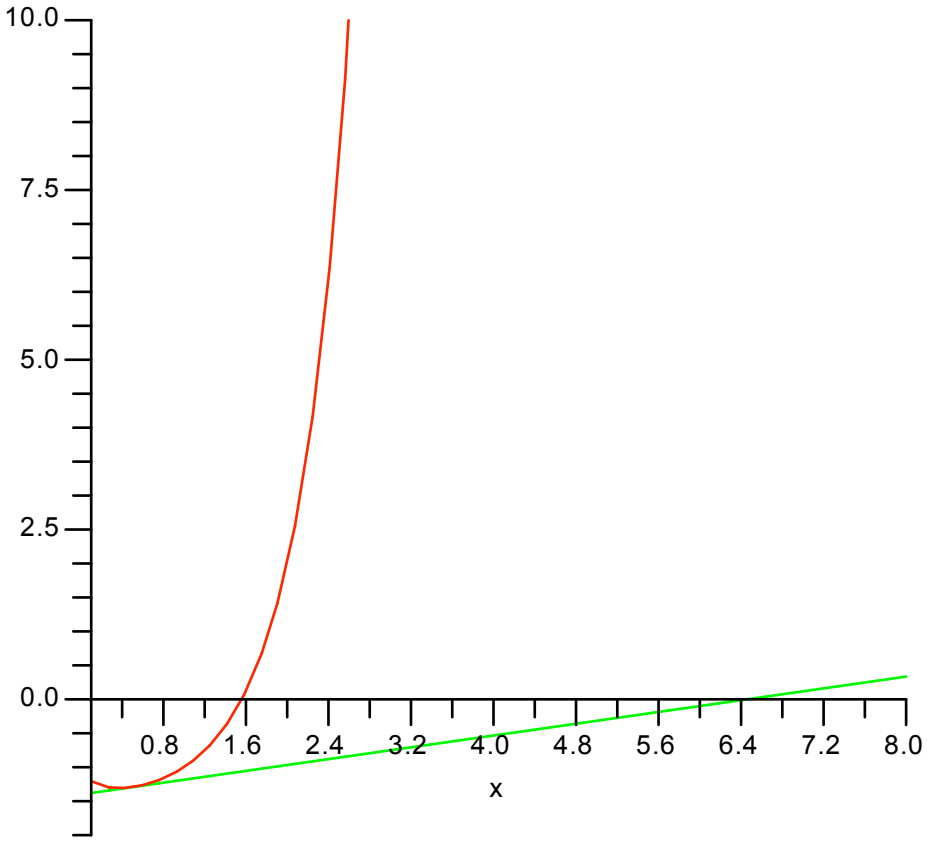
$$t := 1.559610469 \quad (66)$$

Das kann möglicherweise auch eine Weile dauern:

```
> t:=0.5;
```

$$t := 0.5 \quad (67)$$

```
> plot([x^x-2,subs(x=t,f(x))+subs(x=t,f1(x))*(x-t)],x=0.1..8,-2..10)
  ;
```




```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 6.458645349 \tag{68}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 6.109660408 \tag{69}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 5.753783430 \tag{70}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 5.390159079 \tag{71}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 5.017745485 \tag{72}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 4.635274627 \tag{73}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 4.241239606 \tag{74}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 3.834001634 \tag{75}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```
$$t := 3.412299812 \quad (76)$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```
$$t := 2.976967560 \quad (77)$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```
$$t := 2.535883754 \quad (78)$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```
$$t := 2.115736636 \quad (79)$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```
$$t := 1.778292681 \quad (80)$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```
$$t := 1.599662181 \quad (81)$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```
$$t := 1.561090984 \quad (82)$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```
$$t := 1.559612537 \quad (83)$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```
$$t := 1.559610469 \quad (84)$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```
$$t := 1.559610469 \quad (85)$$

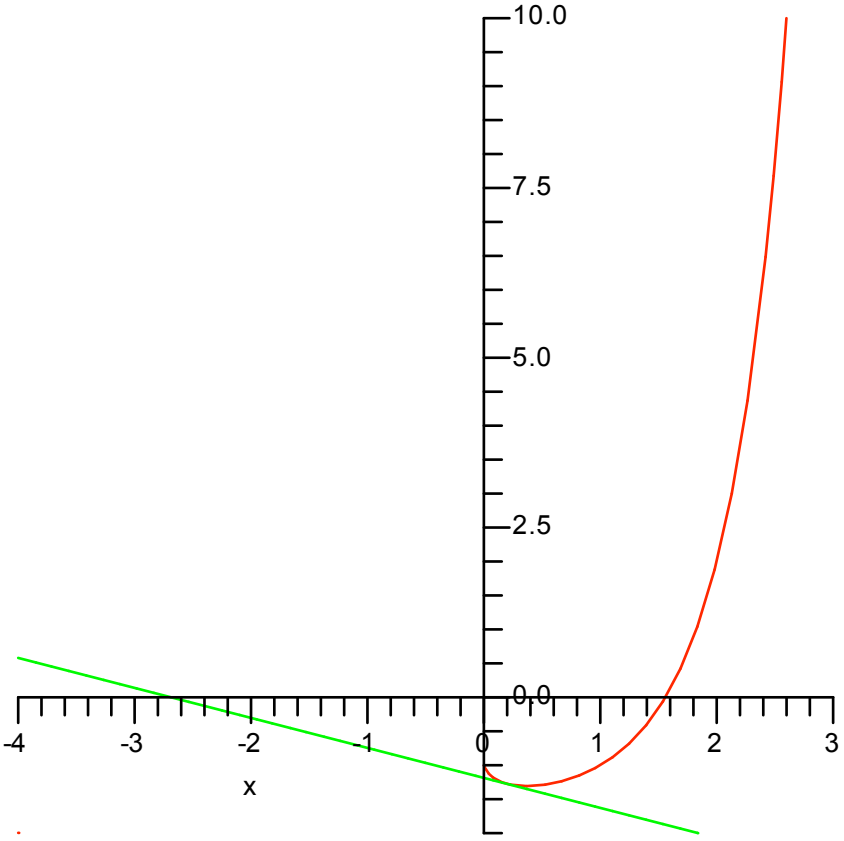
```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```
$$t := 1.559610469 \quad (86)$$

Dafür waren 17 Schritte nötig.

Schliesslich kann ein noch kleinerer Startwert den folgenden Effekt haben:

```
> t:=0.2;
```
$$t := 0.2 \quad (87)$$

```
> plot([x^x-2,subs(x=t,f(x))+subs(x=t,f1(x))*(x-t)],x=-4..3,-2..10);
```




```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := -2.687019806 \tag{88}$$

```
> t := evalf(t-subs(x=t,f(x)/f1(x)));
```

$$t := 0.285025338 + 7.223969321\ I \tag{89}$$

Hier hat t den Definitionsbereich von f(x) verlassen. Das Programm interpretiert den Logarithmus einer negativen Zahl als komplexe Zahl, was für die vorliegende Aufgabe aber nutzlos ist.

>