Lab 11

By: Maegan McGlothin and Justin Peconga

In [1]:
```
def f(x): return sqrt(2*x)
a, b = 1,5
n = 4
```

In [3]:
```
print(list(range(n)))
print([i/2 + 1 for i in range (n)])
```

```
[0, 1, 2, 3]
[1, 3/2, 2, 5/2]
```

In [5]:
```
x_n = [a + (b-a)/n*i for i in range (n+1)]
print(x_n)
```

```
[1, 2, 3, 4, 5]
```

Problem 1

"a" and "b" is the interval. "n" is the number of partinions used to divide the interval. "i" is the value you start with when plugging into the function.

In [6]:
```
fx_n = [f(i).n() for i in x_n]
print(fx_n)
```

```
[1.41421356237310, 2.00000000000000, 2.44948974278318, 2.82842712474619, 3.16227
766016838]
```

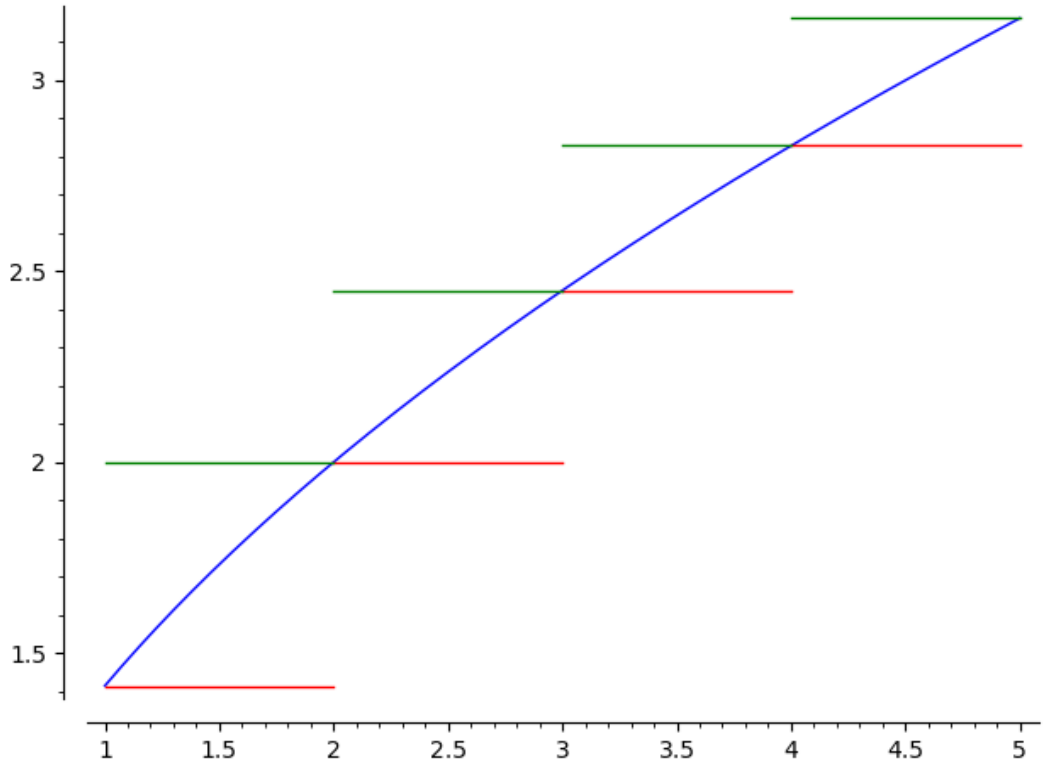
In [7]:
```
print("Left endpoints: ", fx_n[:-1])
print("Right endpoints: ", fx_n[1:])
```

```
('Left endpoints: ', [1.41421356237310, 2.00000000000000, 2.44948974278318, 2.82
842712474619])
('Right endpoints: ', [2.00000000000000, 2.44948974278318, 2.82842712474619, 3.1
6227766016838])
```

In [8]:
```
L = (b-a)/n * sum(fx_n[:-1])
R = (b-a)/n * sum(fx_n[1:])
print("Left Approximation: ", L)
print("Right Approximation: ", R)
```

```
('Left Approximation: ', 8.69213042990246)
('Right Approximation: ', 10.4401945276977)
```

In [9]:
```
P_f = plot(f(x), x, a, b)
P_L = sum([plot(fx_n[i], x, x_n[i], x_n[i+1], color = "red")
          for i in range (n)])
P_R = sum([plot(fx_n[i+1], x, x_n[i], x_n[i+1], color = "green")
           for i in range(n)])
show(P_f + P_L + P_R)
```



Problem 2

Part a)

n=10

In [32]:
```
def f(t): return 100*(1-e^(-0.1*t))
a, b = 1,50
n = 10
```

In [33]:
```
print(list(range(n)))
print([i/2 + 1 for i in range (n)])
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
[1, 3/2, 2, 5/2, 3, 7/2, 4, 9/2, 5, 11/2]
```

In [34]:
```
t_n = [a + (b-a)/n*i for i in range (n+1)]
print(t_n)
```

```
[1, 59/10, 54/5, 157/10, 103/5, 51/2, 152/5, 353/10, 201/5, 451/10, 50]
```

In [35]:
```
t = var('t')
```

In [36]:
```
ft_n = [f(i).n() for i in t_n]
print(ft_n)
```

```
[9.51625819640405, 44.5672715265493, 66.0404474355061, 79.1954817642980, 87.2546
030105179, 92.1918333998847, 95.2165110505802, 97.0695084132959, 98.204703506049
7, 98.9001539824193, 99.3262053000915]
```

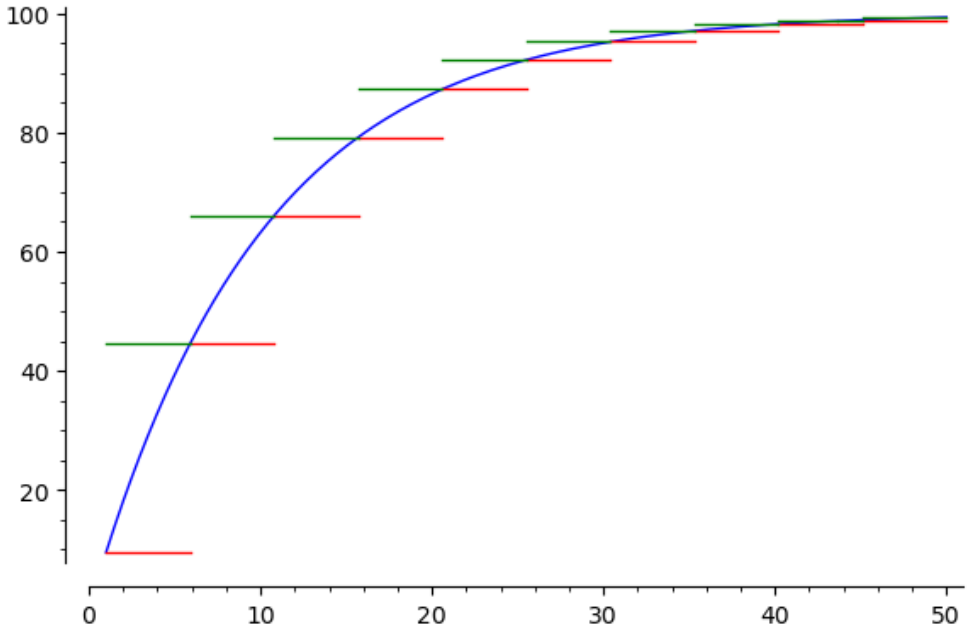
In [37]:
```
print("Left endpoints: ", ft_n[:-1])
print("Right endpoints: ", ft_n[1:])
```

```
('Left endpoints: ', [9.51625819640405, 44.5672715265493, 66.0404474355061, 79.1
954817642980, 87.2546030105179, 92.1918333998847, 95.2165110505802, 97.069508413
2959, 98.2047035060497, 98.9001539824193])
('Right endpoints: ', [44.5672715265493, 66.0404474355061, 79.1954817642980, 87.
2546030105179, 92.1918333998847, 95.2165110505802, 97.0695084132959, 98.20470350
60497, 98.9001539824193, 99.3262053000915])
```

In [38]:
```
L = (b-a)/n * sum(ft_n[:-1])
R = (b-a)/n * sum(ft_n[1:])
print("Left Approximation: ", L)
print("Right Approximation: ", R)
```

```
('Left Approximation: ', 3763.96818419897)
('Right Approximation: ', 4204.03692500704)
```

In [39]:
```
P_f = plot(f(t), t, a, b)
P_L = sum([plot(ft_n[i], t, t_n[i], t_n[i+1], color = "red") for i in range(n)])
P_R = sum([plot(ft_n[i+1], t, t_n[i], t_n[i+1], color = "green") for i in range
(n)])
show(P_f + P_L + P_R)
```



In order to find the amount that it fell we subtracted the left approximation from the right approximation. When using 10 partitions, it fell 440.06874080807 meters.

part b)

n=100

In [40]:
```
def f(t): return 100*(1-e^(-0.1*t))
a, b = 1,50
n = 100
```

In [41]:
```
print(list(range(n)))
print([i/2 + 1 for i in range (n)])
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 2
2, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 4
2, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 6
2, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 8
2, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99]
[1, 3/2, 2, 5/2, 3, 7/2, 4, 9/2, 5, 11/2, 6, 13/2, 7, 15/2, 8, 17/2, 9, 19/2, 1
0, 21/2, 11, 23/2, 12, 25/2, 13, 27/2, 14, 29/2, 15, 31/2, 16, 33/2, 17, 35/2, 1
8, 37/2, 19, 39/2, 20, 41/2, 21, 43/2, 22, 45/2, 23, 47/2, 24, 49/2, 25, 51/2, 2
6, 53/2, 27, 55/2, 28, 57/2, 29, 59/2, 30, 61/2, 31, 63/2, 32, 65/2, 33, 67/2, 3
4, 69/2, 35, 71/2, 36, 73/2, 37, 75/2, 38, 77/2, 39, 79/2, 40, 81/2, 41, 83/2, 4
2, 85/2, 43, 87/2, 44, 89/2, 45, 91/2, 46, 93/2, 47, 95/2, 48, 97/2, 49, 99/2, 5
0, 101/2]
```

In [42]:
```
t_n = [a + (b-a)/n*i for i in range (n+1)]
print(t_n)
```

```
[1, 149/100, 99/50, 247/100, 74/25, 69/20, 197/50, 443/100, 123/25, 541/100, 59/
10, 639/100, 172/25, 737/100, 393/50, 167/20, 221/25, 933/100, 491/50, 1031/100,
54/5, 1129/100, 589/50, 1227/100, 319/25, 53/4, 687/50, 1423/100, 368/25, 1521/1
00, 157/10, 1619/100, 417/25, 1717/100, 883/50, 363/20, 466/25, 1913/100, 981/5
0, 2011/100, 103/5, 2109/100, 1079/50, 2207/100, 564/25, 461/20, 1177/50, 2403/1
00, 613/25, 2501/100, 51/2, 2599/100, 662/25, 2697/100, 1373/50, 559/20, 711/25,
2893/100, 1471/50, 2991/100, 152/5, 3089/100, 1569/50, 3187/100, 809/25, 657/20,
1667/50, 3383/100, 858/25, 3481/100, 353/10, 3579/100, 907/25, 3677/100, 1863/5
0, 151/4, 956/25, 3873/100, 1961/50, 3971/100, 201/5, 4069/100, 2059/50, 4167/10
0, 1054/25, 853/20, 2157/50, 4363/100, 1103/25, 4461/100, 451/10, 4559/100, 1152
/25, 4657/100, 2353/50, 951/20, 1201/25, 4853/100, 2451/50, 4951/100, 50]
```

In [43]:
```
t = var('t')
```

In [44]:
```
ft_n = [f(i).n() for i in t_n]
print(ft_n)
```

```
[9.51625819640405, 13.8430885101042, 17.9630146862169, 21.8859306468624, 25.6212
571979820, 29.1779646532200, 32.5645943759556, 35.7892792912205, 38.859763416759
1, 41.7834204601359, 44.5672715265493, 47.2180019798792, 49.7419774974572, 52.14
52593571159, 54.4336189932297, 56.6125518567009, 58.6872906121782, 60.6628177041
978, 62.5438773224270, 64.3349867947428, 66.0404474355061, 67.6643548750844, 69.
2106088954279, 70.6829227953195, 72.0848323077890, 73.4197040911073, 74.69074381
37500, 75.9010038527476, 77.0533906239093, 78.1506715615237, 79.1954817642980, 8
0.1903303234961, 81.1376063484723, 82.0395847040704, 82.8984314736675, 83.716209
1609803, 84.4948816431281, 85.2363188868447, 85.9423014391673, 86.6145247033852,
87.2546030105179, 87.8640734960990, 88.4443997915776, 88.9969755392000, 89.52312
77388152, 90.0241199346381, 90.5011552496269, 90.9553792747591, 91.387882820145
6, 91.7997045345903, 92.1918333998847, 92.5652111058281, 92.9207343116775, 93.25
92567994572, 93.5815915242996, 93.8885125667412, 94.1807569916614, 94.4590266183
300, 94.7239897058121, 94.9762825577792, 95.2165110505802, 95.4452520882411, 95.
6630549878895, 95.8704427989283, 96.0679135591290, 96.2559414906593, 96.43497813
89187, 96.6054534569157, 96.7677768377918, 96.9223380979710, 97.0695084132959, 9
7.2096412104002, 97.3430730154547, 97.4701242623292, 97.5911000621078, 97.706290
9358071, 97.8159735120571, 97.9204111914191, 98.0198547789371, 98.1145430864412,
98.2047035060497, 98.2905525562467, 98.3722964018467, 98.4501313490961, 98.52424
43170980, 98.5948132866954, 98.6620077278883, 98.7259890068128, 98.786910773258
7, 98.8449193296564, 98.9001539824193, 98.9527473764856, 99.0028258138623, 99.05
05095569373, 99.0959131172866, 99.1391455306724, 99.1803106188896, 99.2195072390
924, 99.2568295211975, 99.2923670939353, 99.3262053000915]
```

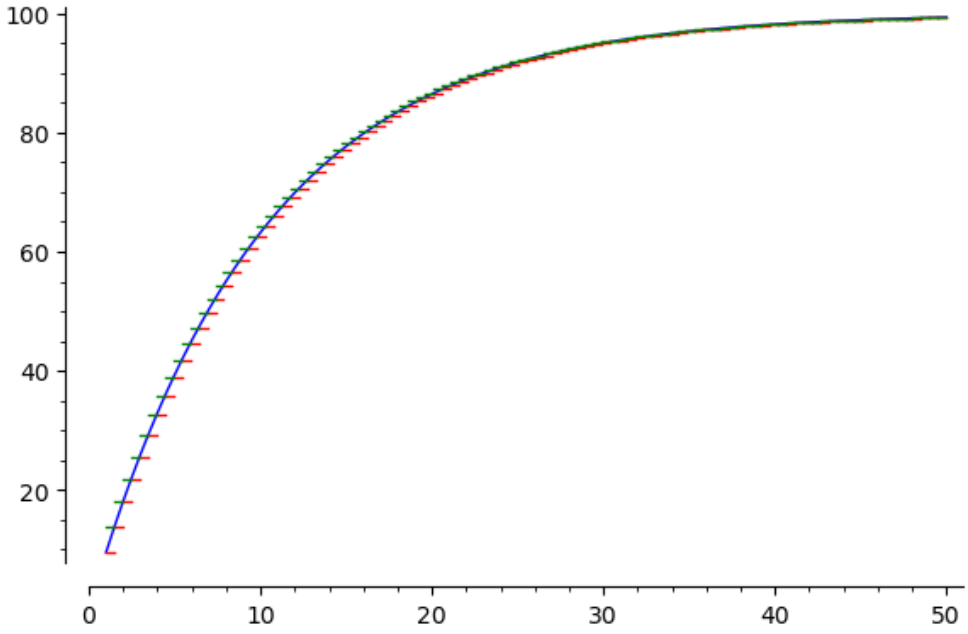
In [45]:
```
print("Left endpoints: ", ft_n[:-1])
print("Right endpoints: ", ft_n[1:])
```

```
('Left endpoints: ', [9.51625819640405, 13.8430885101042, 17.9630146862169, 21.8
859306468624, 25.6212571979820, 29.1779646532200, 32.5645943759556, 35.789279291
2205, 38.8597634167591, 41.7834204601359, 44.5672715265493, 47.2180019798792, 4
9.7419774974572, 52.1452593571159, 54.4336189932297, 56.6125518567009, 58.687290
6121782, 60.6628177041978, 62.5438773224270, 64.3349867947428, 66.0404474355061,
67.6643548750844, 69.2106088954279, 70.6829227953195, 72.0848323077890, 73.41970
40911073, 74.6907438137500, 75.9010038527476, 77.0533906239093, 78.150671561523
7, 79.1954817642980, 80.1903303234961, 81.1376063484723, 82.0395847040704, 82.89
84314736675, 83.7162091609803, 84.4948816431281, 85.2363188868447, 85.9423014391
673, 86.6145247033852, 87.2546030105179, 87.8640734960990, 88.4443997915776, 88.
9969755392000, 89.5231277388152, 90.0241199346381, 90.5011552496269, 90.95537927
47591, 91.3878828201456, 91.7997045345903, 92.1918333998847, 92.5652111058281, 9
2.9207343116775, 93.2592567994572, 93.5815915242996, 93.8885125667412, 94.180756
9916614, 94.4590266183300, 94.7239897058121, 94.9762825577792, 95.2165110505802,
95.4452520882411, 95.6630549878895, 95.8704427989283, 96.0679135591290, 96.25594
14906593, 96.4349781389187, 96.6054534569157, 96.7677768377918, 96.922338097971
0, 97.0695084132959, 97.2096412104002, 97.3430730154547, 97.4701242623292, 97.59
11000621078, 97.7062909358071, 97.8159735120571, 97.9204111914191, 98.0198547789
371, 98.1145430864412, 98.2047035060497, 98.2905525562467, 98.3722964018467, 98.
4501313490961, 98.5242443170980, 98.5948132866954, 98.6620077278883, 98.72598900
68128, 98.7869107732587, 98.8449193296564, 98.9001539824193, 98.9527473764856, 9
9.0028258138623, 99.0505095569373, 99.0959131172866, 99.1391455306724, 99.180310
6188896, 99.2195072390924, 99.2568295211975, 99.2923670939353])
('Right endpoints: ', [13.8430885101042, 17.9630146862169, 21.8859306468624, 25.
6212571979820, 29.1779646532200, 32.5645943759556, 35.7892792912205, 38.85976341
67591, 41.7834204601359, 44.5672715265493, 47.2180019798792, 49.7419774974572, 5
2.1452593571159, 54.4336189932297, 56.6125518567009, 58.6872906121782, 60.662817
7041978, 62.5438773224270, 64.3349867947428, 66.0404474355061, 67.6643548750844,
69.2106088954279, 70.6829227953195, 72.0848323077890, 73.4197040911073, 74.69074
38137500, 75.9010038527476, 77.0533906239093, 78.1506715615237, 79.195481764298
0, 80.1903303234961, 81.1376063484723, 82.0395847040704, 82.8984314736675, 83.71
62091609803, 84.4948816431281, 85.2363188868447, 85.9423014391673, 86.6145247033
852, 87.2546030105179, 87.8640734960990, 88.4443997915776, 88.9969755392000, 89.
5231277388152, 90.0241199346381, 90.5011552496269, 90.9553792747591, 91.38788282
01456, 91.7997045345903, 92.1918333998847, 92.5652111058281, 92.9207343116775, 9
3.2592567994572, 93.5815915242996, 93.8885125667412, 94.1807569916614, 94.459026
6183300, 94.7239897058121, 94.9762825577792, 95.2165110505802, 95.4452520882411,
95.6630549878895, 95.8704427989283, 96.0679135591290, 96.2559414906593, 96.43497
81389187, 96.6054534569157, 96.7677768377918, 96.9223380979710, 97.069508413295
9, 97.2096412104002, 97.3430730154547, 97.4701242623292, 97.5911000621078, 97.70
62909358071, 97.8159735120571, 97.9204111914191, 98.0198547789371, 98.1145430864
412, 98.2047035060497, 98.2905525562467, 98.3722964018467, 98.4501313490961, 98.
5242443170980, 98.5948132866954, 98.6620077278883, 98.7259890068128, 98.78691077
32587, 98.8449193296564, 98.9001539824193, 98.9527473764856, 99.0028258138623, 9
9.0505095569373, 99.0959131172866, 99.1391455306724, 99.1803106188896, 99.219507
2390924, 99.2568295211975, 99.2923670939353, 99.3262053000915])
```

In [46]:
```
L = (b-a)/n * sum(ft_n[:-1])
R = (b-a)/n * sum(ft_n[1:])
print("Left Approximation: ", L)
print("Right Approximation: ", R)
```

```
('Left Approximation: ', 3979.71740437727)
('Right Approximation: ', 4023.72427845807)
```

In [83]:
```
P_f = plot(f(t), t, a, b)
P_L = sum([plot(ft_n[i], t, t_n[i], t_n[i+1], color = "red") for i in range(n)])
P_R = sum([plot(ft_n[i+1], t, t_n[i], t_n[i+1], color = "green") for i in range
(n)])
show(P_f + P_L + P_R)
```



In order to find the number of meters that it fell at 100 partitions, we subtracted the left approximation from the right approximation. When using 100 partitions, it fell 44.0068740808 meters.

part c)

In [139]:
```
def f(t): return 100*(1-e^(-0.1*t))
a, b = 1,50
n = 100000000
```

In [140]:
```
t = var('t')
```

In [141]:
```
L = (b-a)/n * sum(ft_n[:-1])
R = (b-a)/n * sum(ft_n[1:])
print("Left Approximation: ", L)
print("Right Approximation: ", R)
```

```
('Left Approximation: ', 4.00187852534641)
('Right Approximation: ', 4.00192253222049)
```

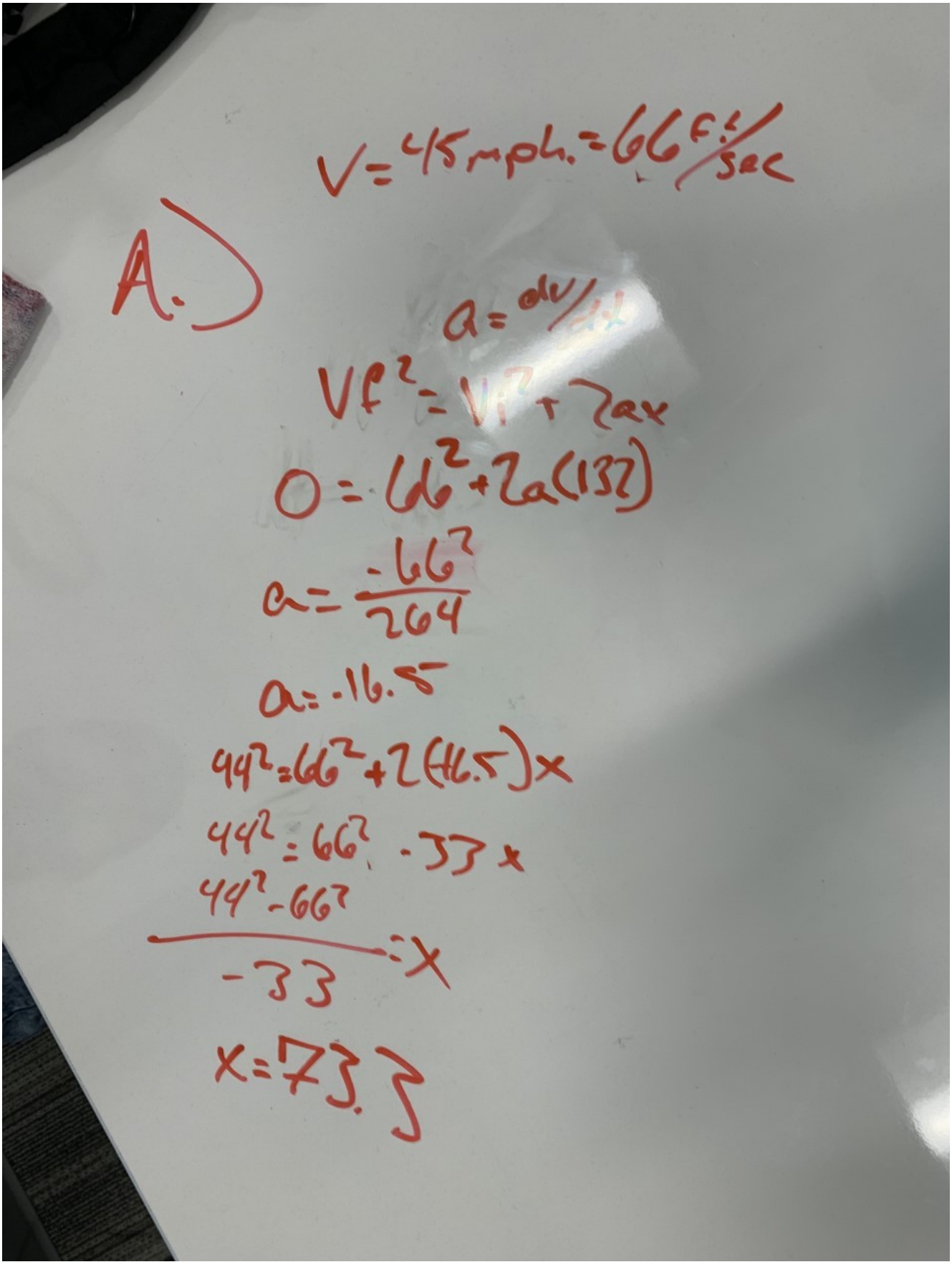
In [142]:
```
R-L
```

Out[142]:
```
0.0000440068740807931
```

To solve part c, we started with 100 in the previous step and knew that was too far away from our amount of error so we then began to add more zeros. We repeated this process until we were within the area of error. In order to get R and L to differ by 0.01 we need to have n equal 10000000.

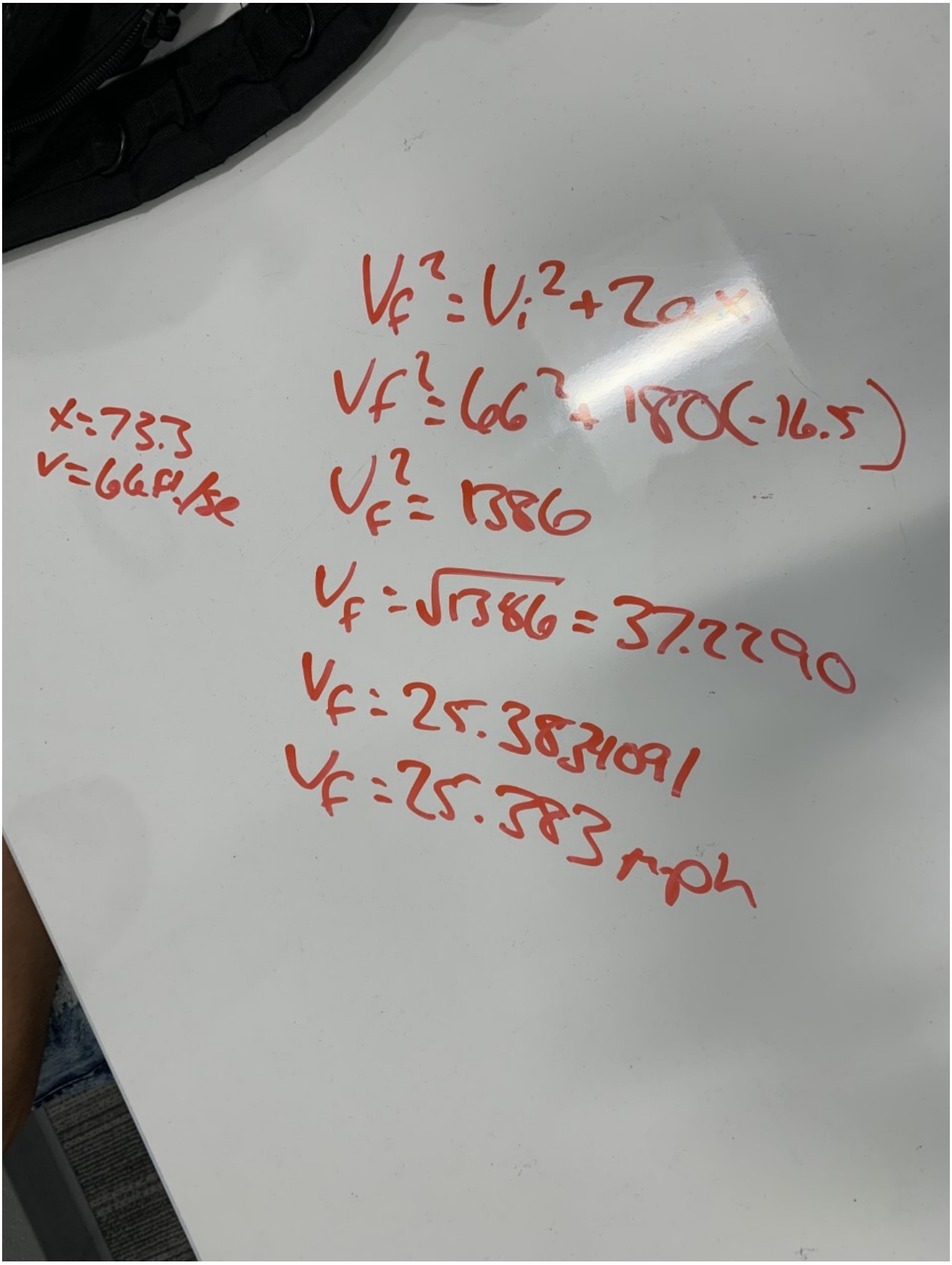
Problem 3

part a)


V = 45 mph = 66 ft/sec
A.)
a = dv/dt
Vf² = Vi² + 2ax
0 = 66² + 2a(132)
a = -66²/264
a = -16.5
44² = 66² + 2(16.5)x
44² = 66² - 33x
44² - 66² / -33 = x
x = 73.5

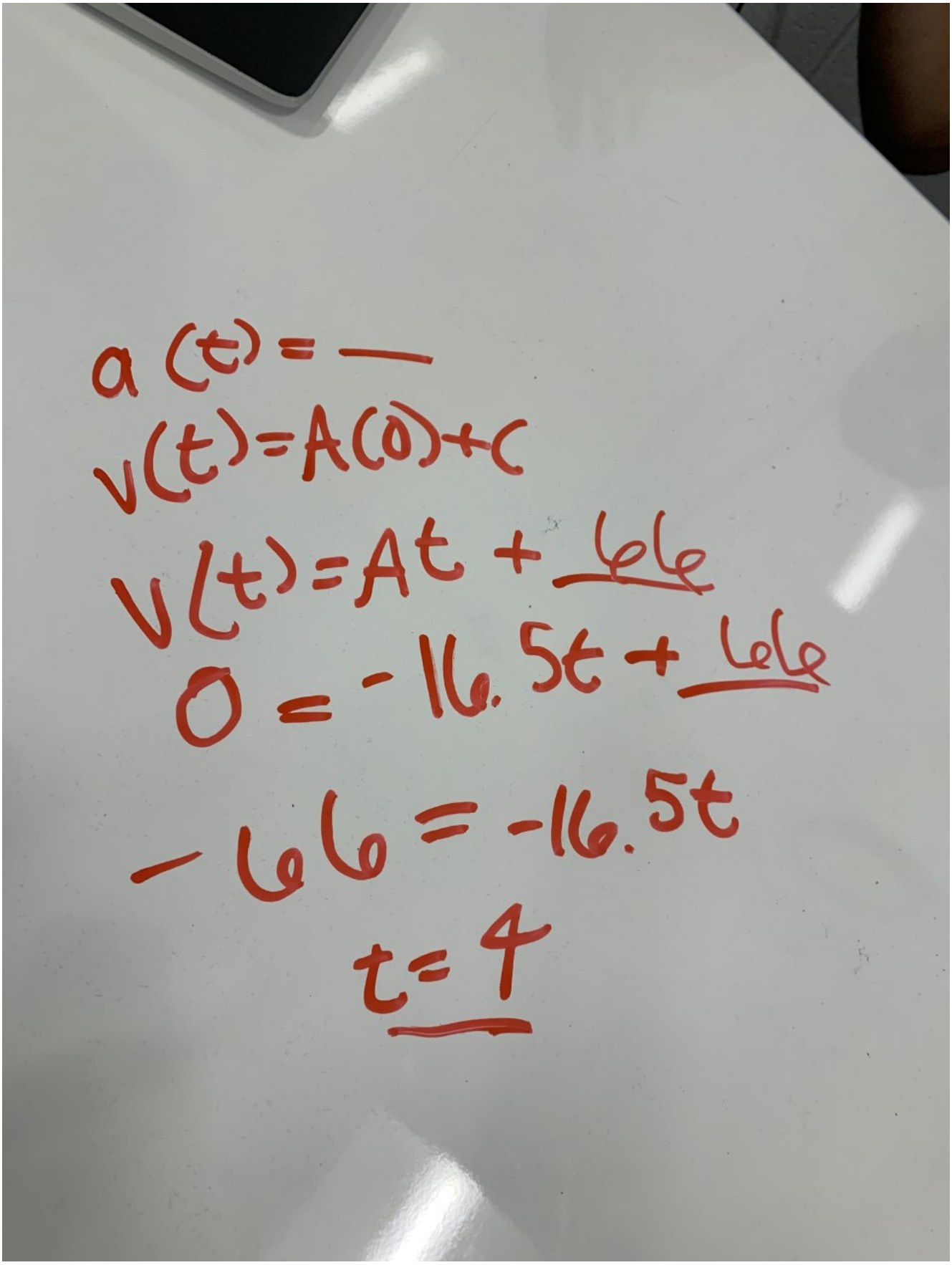
Using the first formula we were able to solve for a. Then we could plug the value for a into the equation along with the velocity in order to find the distance the car traveled when the velocity was 30mph. We found that the car had traveled 73.3 ft.

part b)


$V_f^2 = V_i^2 + 2ax$
$V_f^2 = 66^2 + 180(-16.5)$
$V_f^2 = 1386$
$V_f = \sqrt{1386} = 37.22290$
$V_f = 25.38341091$
$V_f = 25.383$ mph
x = 73.3
v = 66 ft/sec

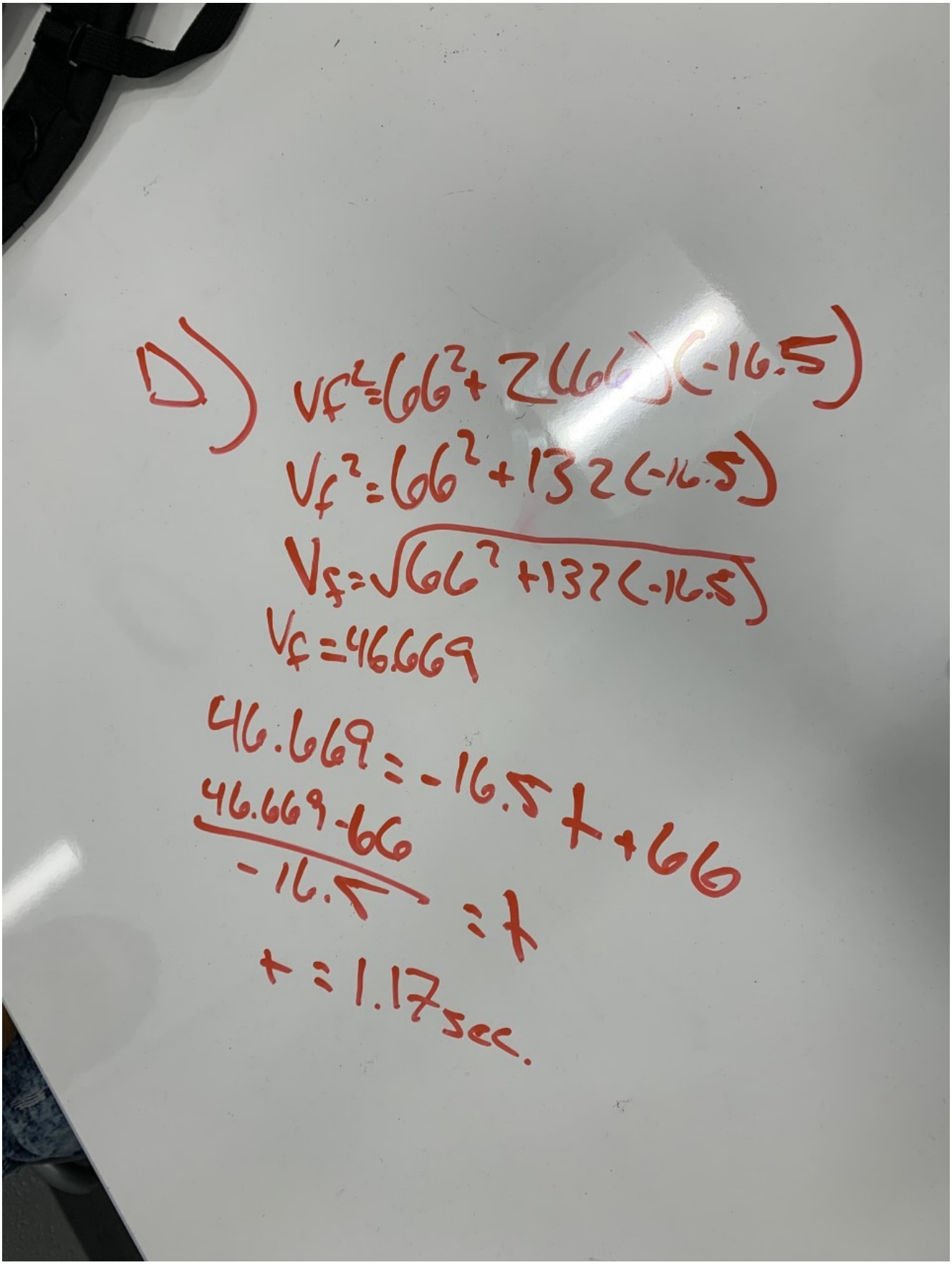
When solving for the distance at 90ft we were able to use a lot of information from the preiovus step in order to solve for vf at 900 ft. Once solved we found that the car was going 25.383 mph after 90ft.

part c)


a(t) = —
v(t) = A(0) + C
v(t) = At + 66
0 = -16.5t + 66
-66 = -16.5t
t = 4

In order to solve to find how long it takes the car to stop we need to set the position function equal to 0. Once we set the function equal to 0 we can just solve algebraically. Once solved, we found out that it takes the car 4 seconds to stop.

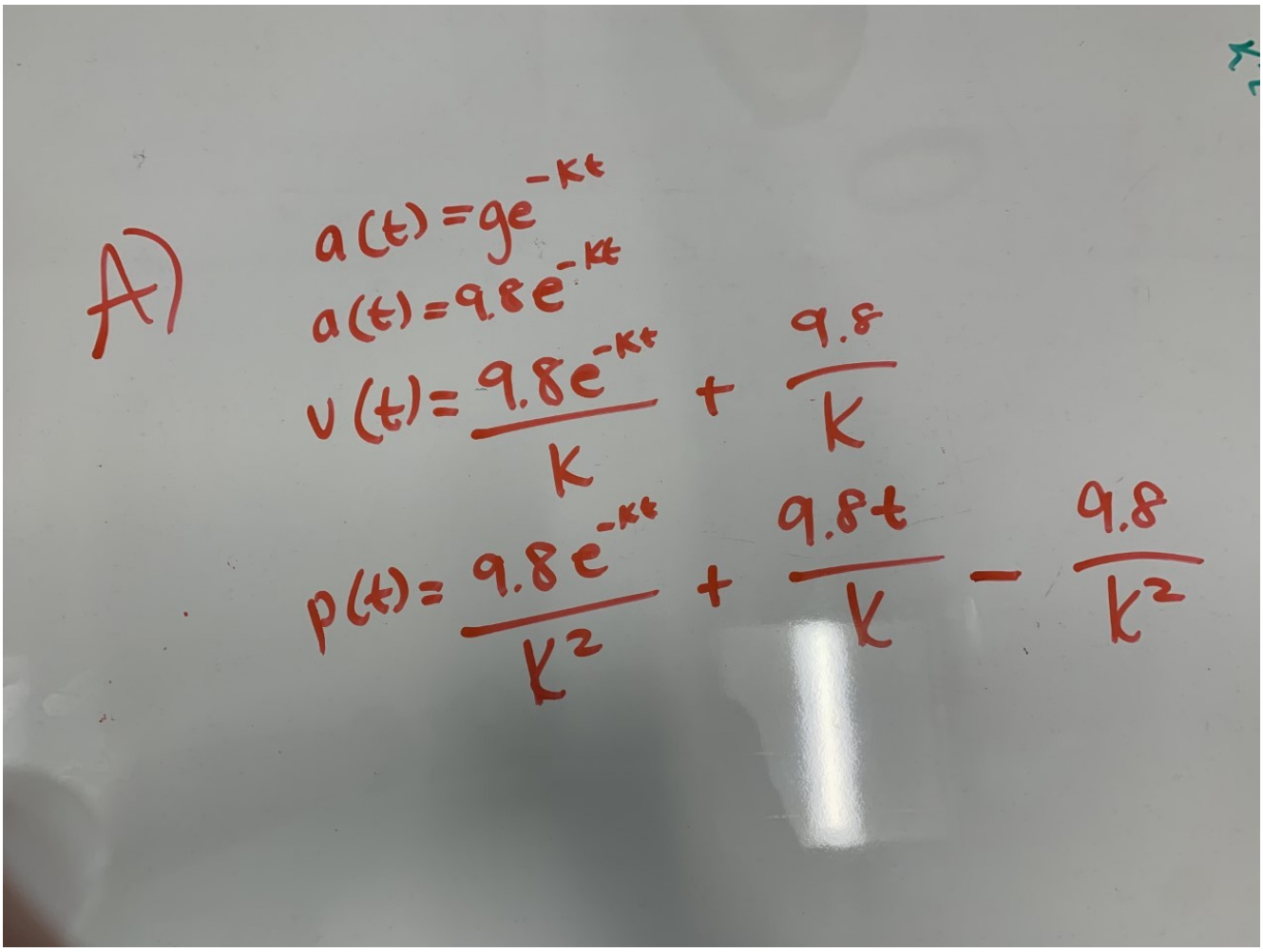
part d)


D) $v_f^2 = 66^2 + 2(66)(-16.5)$
$v_f^2 = 66^2 + 132(-16.5)$
$v_f = \sqrt{66^2 + 132(-16.5)}$
$v_f = 46.669$
$46.669 = -16.5t + 66$
$\frac{46.669 - 66}{-16.5} = t$
$t = 1.17$ sec.

Since we were searching for t at 1/2d we were able to plug in half the distance with the rest of our information from the previous steps in order to find t at 1/2d. We found that it takes the car 1.17 seconds to reach the halfway point.

Problem 4

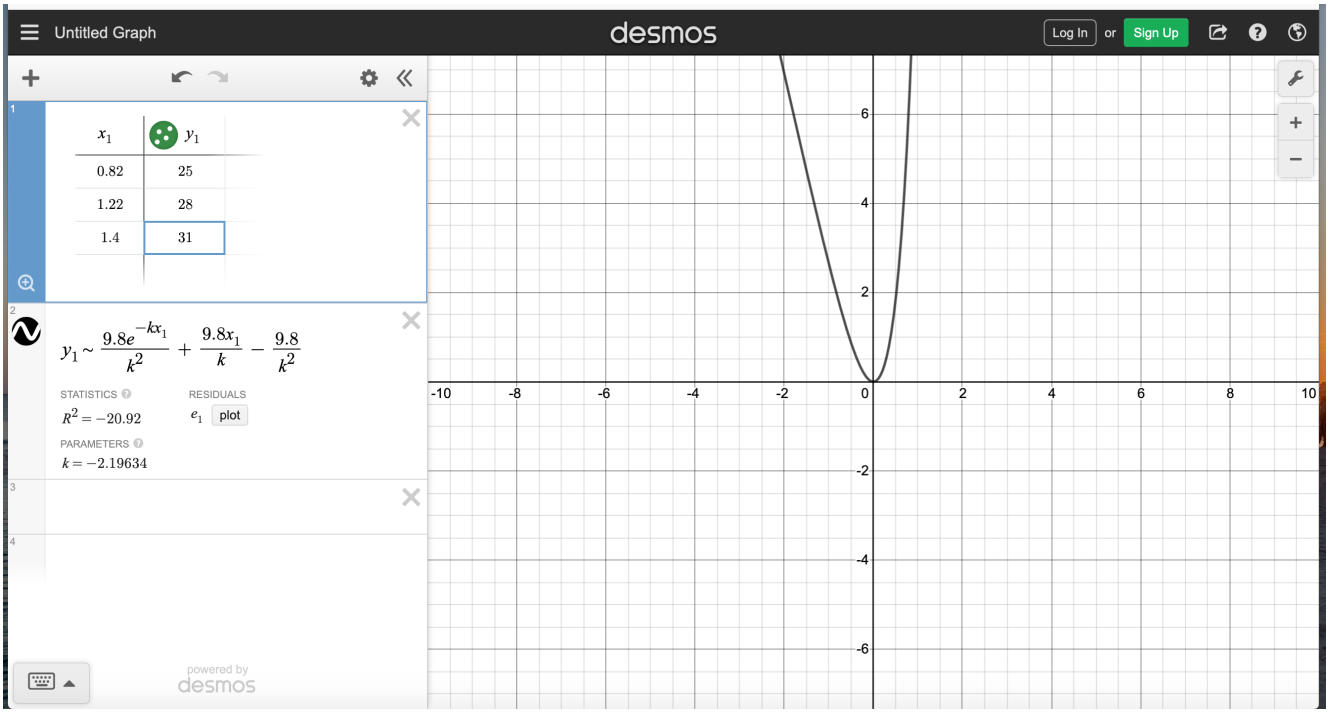
part a)



In part a, we began with the given accelartion function and took the derivate once to give us the velocity function and then took the derivate a second time to give us the position function. This poistion function would then be used in the following steps.

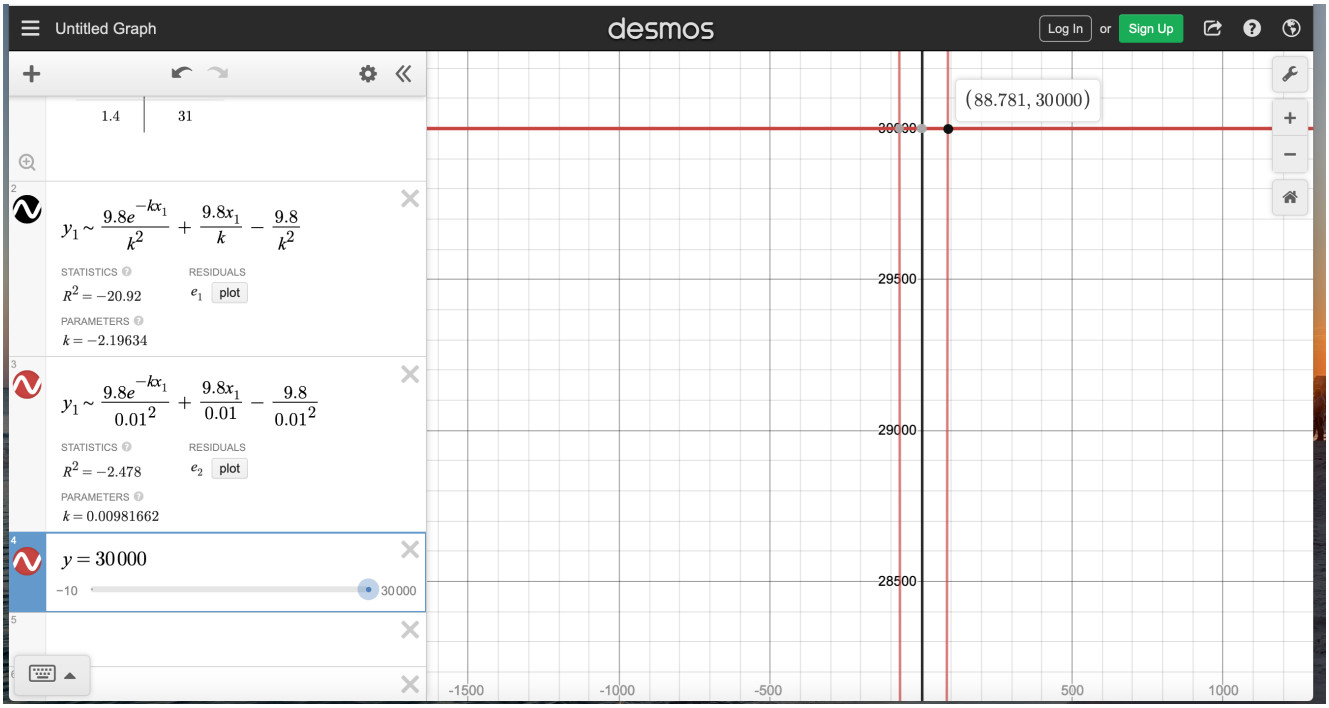
part b)

We dropped the tennis ball from 3 different heights in the stairwell and then reported them into a table in desmos. By plotting our points in desmos and then plotting our position function we were able to calculate the value of k.



However our data that we collected did not match reality, therefore we resulted to using 0.01 as our k value.

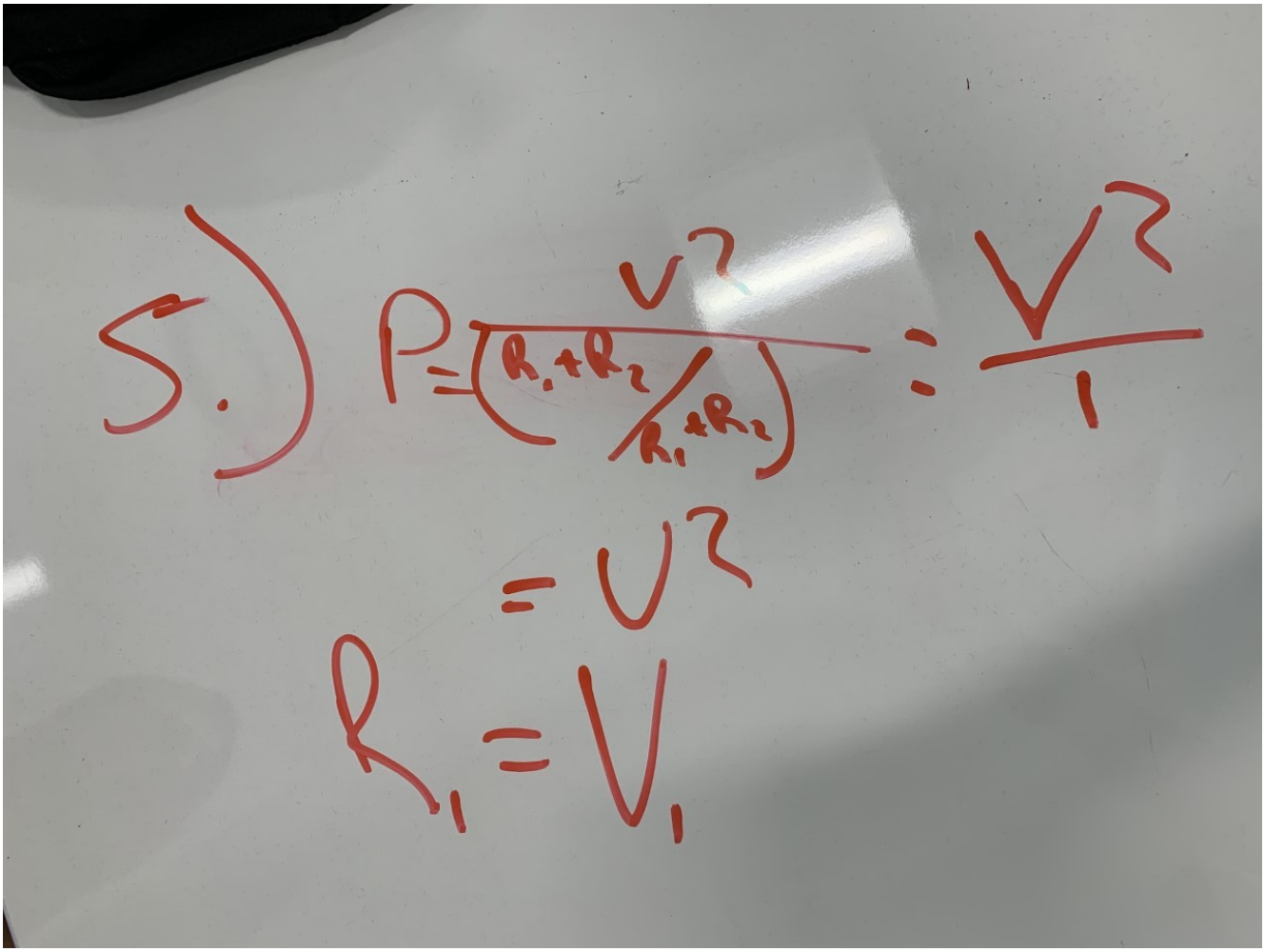
part c)



In order to solve part c we first plugged 0.1 in for k and then plotted that function on desmos. Next, we plotted y=30,000 to find how long it would take a tennis ball dropped from the airplane at 30000 feet. We zoomed out until we found the point of intersection and that gave us the answer of 88.781.

It would take a tennis ball dropped from 30000 feet 88.781 seconds to hit the ground.

Problem 5



Since we are trying to find the max power. We set r1 and v1 to be equal to each other. Doing so allows us to solve the equation given in the problem so we were able to simplify the equation to v^2.