Midterm

October 27, 2019

1 Problem 1: Nano XIX Sneakers

[1]:
```
import matplotlib.pyplot as plt
import numpy as np
from numpy import matrix
```

1.0.1 Cost function

c(x) = 18x + 24000

1.0.2 Revenue function

r(x) = 120x

1.0.3 Breakeven point

120x = 18x + 24000 102x = 24000 x = 24000/102

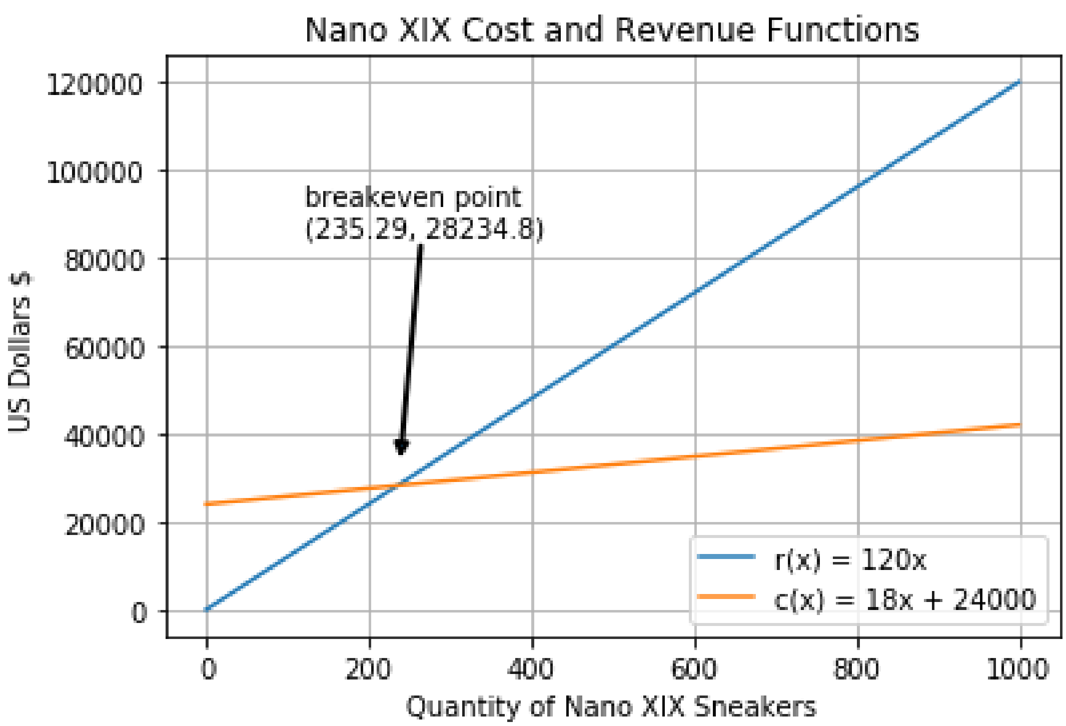
r(x) = 120x

r(24000/102) = 120(24000/102)

y = 120*(24000/102) or approx. $28,235.29

[2]:
```
def graph(x_range, BE_X=None, BE_Y=None):
    x = np.array(x_range)
    rev_y = revenue_function(x)
    cost_y = cost_function(x)
    plt.plot(x, rev_y)
    plt.plot(x, cost_y)
    plt.legend (('r(x) = 120x','c(x) = 18x + 24000'),loc=4)
    plt.xlabel("Quantity of Nano XIX Sneakers")
    plt.ylabel("US Dollars $")
    plt.title("Nano XIX Cost and Revenue Functions")
    if BE_X == None:
        BE_X = round(24000/102, 2)
    if BE_Y == None:
        BE_Y = round(120*BE_X, 2)
    plt.annotate('breakeven point\n({}, {})'.format(BE_X, BE_Y), xy=(BE_X,␣
 ↪BE_Y), xytext=(120, 85000),
```

```
                 arrowprops=dict(
                          connectionstyle="arc3,rad=0.",
                          shrinkA=0, shrinkB=10,
                          arrowstyle= '-|>', ls= '-', linewidth=2
                          ))
    plt.grid()
    plt.show()

def revenue_function(x):
    return 120*x

def cost_function(x):
    return 18*x + 24000

graph(range(0, 1000))
```



1.1 Mathematical Breakeven Point is (235.29, 28,234.80)

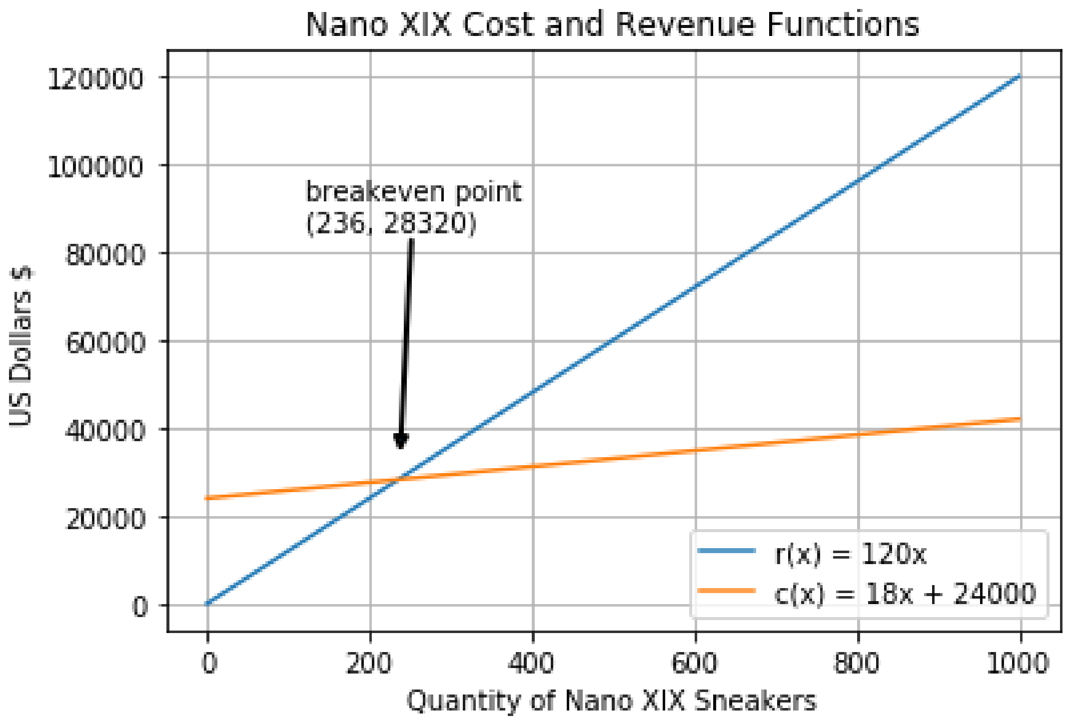
where cost and revenue are equal at $28,234.80 and number of sneakers produced is 235.29, mathematically.

Since one cannot have a partial sneaker, assuming no hungry dogs were involved, we round up to 236.

r(236) = 120*236 = $28,320

1.2 Practical Breakeven point (236, 28320)

[3]:
```
graph(range(0, 1000), BE_X=236, BE_Y=28320)
```



2 Problem 2: Arielle's Investments

The following equations apply to Arielle's investments, where a is mutual funds, b is government bonds, and c is CD's:

a + b + c = 17500

1.11a + 1.07 + 1.05c = 18995

a = 2c

Written as matrix:

1 1 1 | 17500

1.11 1.07 1.05 | 18995

-1 0 2 | 0

Using TI84 Calculator rref([A]) where A is the matrix above, the resulting matrix is below:

1 0 0 | 9000

0 1 0 | 4500

0 0 1 | 4500

a = 9000 b = 4000 c = 4500

Plugging into the originals to double check:

9000 + 4000 + 4500 = 17500

2(4500) = 9000

1.11(9000) + 1.07(4000) + 1.05(5400) = 18995

2.1 Therefore, Arielle invested 9000 USD in mutual funds, 4000 USD in government bonds, and 4500 USD in CDs.

3 Problem 3: Vandelay Industries

a + b + c + d = 252

a = 3b

c = 2d

where a is the first team, b is the second team c is the third, and d is the fourth team.

Augmented matrix:

a b c d |

1 1 1 1 | 252

1 -3 0 0 | 0

0 0 1 -2 | 0

Using TI-84 graphing calculator with 3x5 matrix A and the amazing function rref([A]), the following is resulting matrix:

a b c d |

1 0 0 9/4 | 189

0 1 0 3/4 | 63

0 0 1 -2 | 0

Since there are four variables and only 3 equations, this is normal.

The resulting matrix can tell us about the distribution in terms of the fourth team, d.

a = 189 - 9/4d

b = 63 - 3/4d

c = 2d (same as one of the original equations)

We also still have the original equation a = 3b

3.1 The third team is twice the fourth team. The first team is thrice the third team, or the first team is 189 less 9/4 of the fourth team. The second team is 63 less 3/4 the fourth team.

4 Problem 4: Phil's Candy

[4]:
```
candy = matrix([[6,8,1],[6,4,1],[5,7,1]])
print (candy)
```

```
[[6 8 1]
 [6 4 1]
 [5 7 1]]
```

[5]:
```
K = matrix([[3],[5],[2]])
print(K)
```

```
[[3]
 [5]
 [2]]
```

[6]:
```
cost_of_one_batch = np.dot(candy,K)
cost_of_100_batches = np.dot(cost_of_one_batch, 100)
print("cost of 100 batches using supplier K : \n{}".format(cost_of_100_batches))
print("\ntotal cost : {}".format(sum(cost_of_100_batches)))
```

```
cost of 100 batches using supplier K :
[[6000]
 [4000]
 [5200]]

total cost : [[15200]]
```

##

cost of 100 batches using supplier K in $

chocolate bar type	cost
cherry	6000
almond	4000
raisen	5200

5 Problem 5: Welsh-Ryan Arena

Where c is courtside, f is first level, and u is upper deck:

c + f + y = 15000 8c + 6f + 4u = 76000 4c + 6f + 2u = 44000

Written as an Augmented matrix:

1 1 1 | 15000

8 6 4 | 76000

4 6 2 | 44000

Using a Ti-84 calculator rref([A]) where A is the above matrix, the resulting matrix is below:

1 0 0 | 3000

0 1 0 | 2000

0 0 1 | 10000

c = 3000

f = 2000

u = 10000

This makes sense from a glance that upper deck seats are the most plentiful. We can be sure by plugging these values into the original equations:

3000 + (2000) + 10000 = 15000 is correct

8(3000)+ 6(2000) + 4(10000) = 76000 is correct

4(3000) + 6(2000) + 2(10000) = 44000 is also correct

5.1 Therefore, there are 3000 courtside seats, 2000 first level, and 10,000 upper deck seats in the stadium.

6 Problem 6: Hoch Industries

	old	new
quantity of processes	x	y
sulphur	6x	2y
lead	3x	4y

6x + 2y <= 18000

3x + y <= 9000

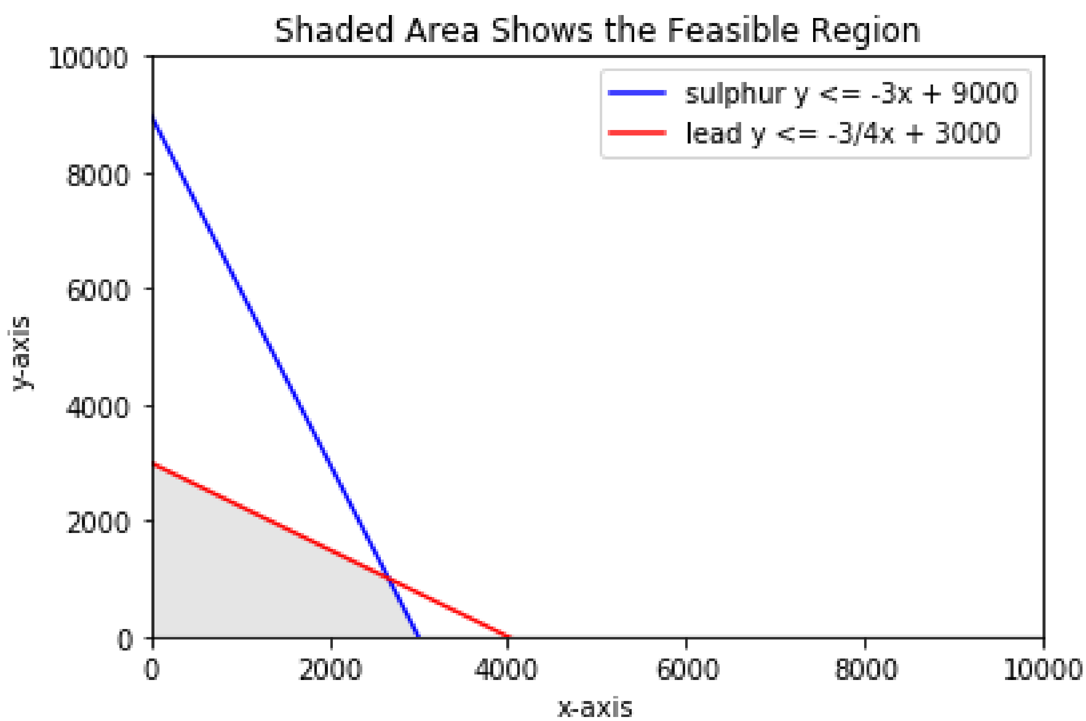
z = 25x + 16y

Graphing y <= -3x + 9000 for sulphur and y <= -3/4x + 3000 for lead:

[7]:
```
x = np.arange(0, 100000, 10)

y_sulphur = -3*x + 9000

y_lead = -3/4*x + 3000
```

```
y3 = np.minimum(y_sulphur, y_lead) # the smaller of sulphur and lead functions,␣
↪since both have y<=
```

[8]:
```
plt.xlim(0, 10000)
plt.ylim(0, 10000)
plt.xlabel('x-axis')
plt.ylabel('y-axis')
plt.title('Shaded Area Shows the Feasible Region')
plt.plot(x, y_sulphur, color='b', label='sulphur y <= -3x + 9000')
plt.plot(x, y_lead, color='r', label='lead y <= -3/4x + 3000')
plt.legend()
plt.fill_between(x, 0, y3, color='grey',alpha=0.2)
plt.show()
```



corner points are where lead function hits the y-axis, where lead and sulphur functions meet, and where sulphur function hits the x-axis.

lead   y <= -3/4x + 3000

sulphur y <=   -3x + 9000

-3/4x + 3000 = -3x + 9000 x = 6000/2.25 ~= 2667 y = -3(6000/2.25) + 9000 = 1000

Corner Points	z = 25x+16y
(0, 0)	0

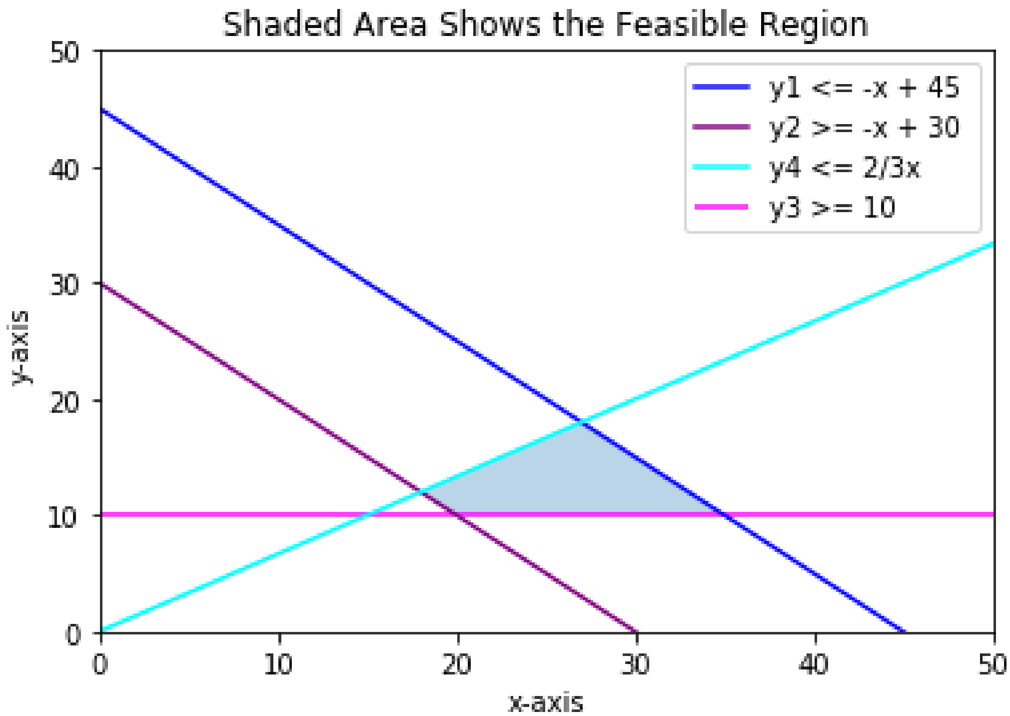
Corner Points	z = 25x+16y
(0, 3000)	48000
(6000/2.25, 1000)	82667
(3000, 0)	75000

The point where the two functions meet give the maximum value.

6.1 Hoch Industries should run 2667 of the old and 1000 of the new.

7 Problem 7: Northwestern

[9]:
```
x = np.arange(0, 1000, 1)

y1 = -x + 45
y2 = -x + 30
y3 = [10 for i in range(0, 1000)]
y4 = (2/3)*x

plt.xlim(0, 50)
plt.ylim(0, 50)
plt.xlabel('x-axis')
plt.ylabel('y-axis')
plt.title('Shaded Area Shows the Feasible Region')
plt.plot(x, y1, color='b', label='y1 <= -x + 45 ')
plt.plot(x, y2, color='purple', label='y2 >= -x + 30')
plt.hlines(10, 0, 50, color='magenta', label='y3 >= 10')
plt.plot(x, y4, color='cyan', label='y4 <= 2/3x')
plt.legend()
x_fill = [20, 18, 27, 35]
y_fill = [10, 12, 18, 10]
plt.fill_between(x_fill, y_fill, 10, alpha=0.3)
plt.show()
```



	z = 2400x + 1100y
(20,10)	59,000
(18,12)	56,400
(27,18)	84,600
(35,10)	95,000

7.1 Northwestern should staff 18 teachers and 12 TA's for a minimized cost of $56,400.

8 Problem 8: Roger the Athlete

	pill 1	pill 2	pill 3	mins
qty of pills	y1	y2	y3	-
A	4y1	1y2	10y3	>=10
B	3y1	2y2	1y3	>=12
C	0y1	4y2	5y3	>=20

Mimimize cost: z = 0.06y1 + 0.08y2 + 0.01y3

4y1 + y2 + 10y3 >= 10

3y1 + 2y2 + y3 >= 12

4y2 + 5y3 >= 20

y1 >= 0

y2 >= 0

y3 >= 0

We cannot graph these… but we can attempt a matrix. Since there are 3 equations, there will be 3 slack variables. A minimization problem is also just a maximization problem with the objection function z = -w

z = -0.06y1 - 0.08y2 - 0.01y3

y1 y2 y3 s1 s2 s3 z |

4 3 0 1 0 0 0 | 0.06

1 2 4 0 1 0 0 | 0.08

10 1 5 0 0 1 0 | 0.01

-10 -12 -20 0 0 0 1 | 0

-10 is the least negative so will be the pivot column.

[10]:
```python
print(.06/4)
print(.08/1)
print(.01/10)
```

```
0.015
0.08
0.001
```

10 will be the pivot because it's the smallest quotient.

Resulting matrix comes from the following row transformations:

-4/10R3 + R1 -> R1

-1/10R3 + R2 -> R2

R3 + R4 -> R4

on a Ti-84 calculator, where [B] is the Matrix shown above:

*row+(-2/5, [B], 3, 1) -> [C]

*row+(-1/10, [C], 3, 2) -> [D]

*row+(1, [D], 3, 4) -> [E]

y1 y2 y3 s1 s2 s3 z |

0 2.6 -2 1 0 -0.4 0 | 0.056

0 1.9 1 -11 1 0 0 | 0.08

10 1 5 0 0 1 0 | 0.01

---

0 -11 -15 0 0 1 1 | 0.01

-11 is the least negative so will be the next pivot column.

[11]:
```
print(0.056/2.6)
print(0.08/1.9)
print(0.01/1)
```

```
0.021538461538461538
0.04210526315789474
0.01
```

1 will be the pivot, in the third row.

Resulting matrix comes from the following row transformations:

-2.6/1R3 + R1 -> R1

-1.9/1R3 + R2 -> R2

11/1R3 + R4 -> R4

on a Ti-84 calculator, where [B] is the Matrix shown above:

*row+(-2.6, [E], 3, 1) -> [F]

*row+(-1.9, [F], 3, 2) -> [G]

*row+(11, [G], 3, 4) -> [H]

y1 y2 y3 s1 s2 s3 z |

-26 0 -15 1 0 -3 0 | 0.03

-19 0 -6 0 1 -2 0 | 0.06

10 1 5 0 0 1 0 | 0.01

---

110 0 40 0 0 12 1 | 0.12

There are no more negative indicators, so the optimum solution has been achieved.

y1 y2 y3 s1 s2 s3 z |

-2600 0 -1500 100 0 -300 0 | 3

-1900 0 -600 0 100 -200 0 | 6

1000 100 500 0 0 100 0 | 1

11000 0 4000 0 0 1200 100 | 12

Basic variables from this Tableau are y2, s1, and s2.

$y1 = 0$

$y2 = 1$

$y3 = 0$

$s1 = 3$

$s2 = 6$

$s3 = 0$

and $z = 12$

9 Problem 9: Better Buy Inventory

Better Buy has a capacity of 210 items. The fact that they don't take the size of the items into account is something I wouldn't point out if I was their consultant, because it makes this problem simpler.

x1: DVD players

x2: surround sound stereo speakers

x3: smart tvs

$x1 + x2 + x3 = 210$

$2x2 = x1$

$x3 >= 30$

$z = 450x1 + 2000x2 + 750x3$

I notice x3 has to be greater than or equal to 30, but it's also kind of cheap.

Based on x2's relationship with x1, and taking a shot at $x3 = 30$ since it's less than half the price of x2, we can rewrite the first equation:

$2x2 + x2 + 30 = 210$

$3x2 = 180$

$x2 = 60$ implying $x1 = 120$

$z = 450(120) + 2000(60) + 750(30) = \$196{,}500$

if x3 is a little larger, it makes the numbers not round out at 31 and 32, until it gets to 33. So if $x3 = 33$, $x2 = 59$, and $x1 = 118$, and profit is lower at \$195,850.

I wouldn't use this kind of deductive reasoning in every situation, but 210 is not a very high number, so I don't mind shooting darts until I reach the answer, kind of like neural networks do.

I did some iterations on the calculator and it looked linear that the higher x3 got, the lower the profit, but just to be safe, I decided to write a short python program (below), finding it is safe to conclude the following:

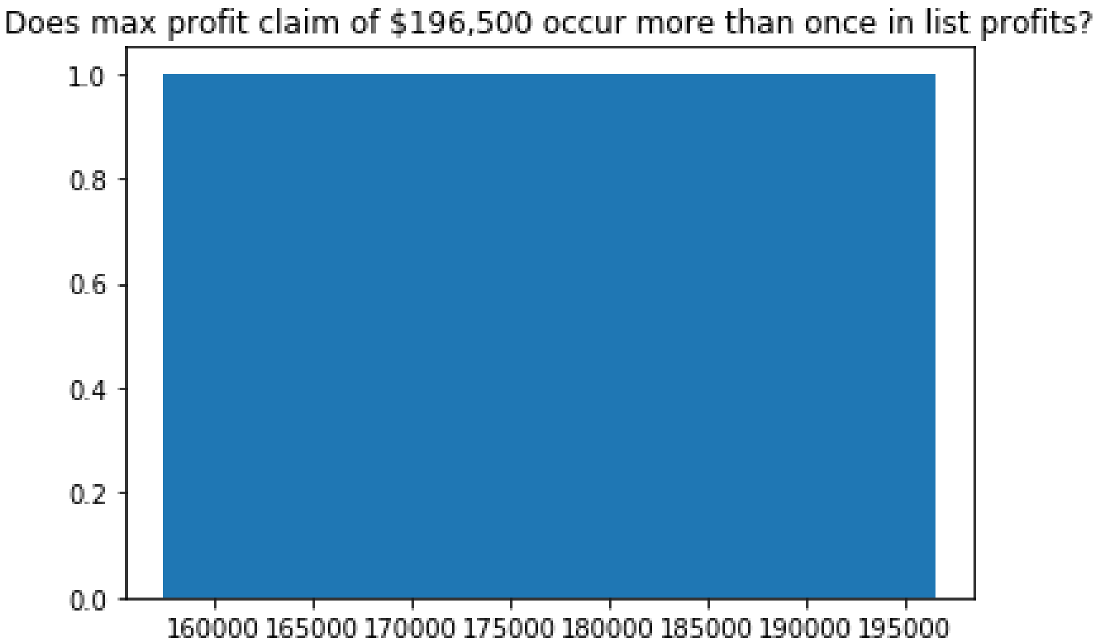
9.1 120 DVD players, 2000 surround sound stereo systems, and 30 smart TV's yields the maximum profit of $196,500

[12]:
```
profits = []
all_the_info = []

def find_x2(x3):
    return (210 - x3)/3

def find_x1(x2):
    return 2*x2

def find_z(x1, x2, x3):
    return 450*x1 + 2000*x2 + 750*x3

for x3 in range(30, 211):
    x2 = find_x2(x3)
    x1 = find_x1(x2)
    z = find_z(x1, x2, x3)
    profits.append(z)
    all_the_info.append({z:[x1,x2,x3]})
```

[13]:
```
max(profits)
```

[13]:
```
196500.0
```

[14]:
```
plt.hist(profits, bins=len(profits))
plt.title("Does max profit claim of $196,500 occur more than once in list␣
 ↪profits?")
plt.show()
```



Since 196,500 only occurs once in the profits list, this conclusion is valid.

10 Problem 10: McDowell's Sweepstakes

P(A|winner) = P(A) * P(winner|A) / (P(A) * P(winner|A) + P(B) * P(winner|B))

P(A) = 0.27

P(A') = P(B) = 1 - 0.27 = 0.73

P(winner|A) = 0.04

P(winner|B) = 0.05

P(A|winner) = [(0.27) * (0.04)] / [(0.27) * (0.04) + (0.73) * (0.05)]

10.1 The probability of a winner coming from box A is 22.83%

[15]:
```
num = 0.27*0.04
b = 0.73*0.05

print("{} %".format(round(num/(num+b)*100, 2)))
```

```
22.83 %
```