In [62]:

```
import numpy as np
import pandas as pd
from scipy import stats
import matplotlib.pyplot as plt
import itertools
```

Problem 1

1. Suppose we have any array Z = [0, 1, 2, . . . , 255].

> (a) Form this array without typing out all of the numbers.
>
> (b) Reshape this array to be a 16 × 16 matrix.
>
> (c) Determine Z1, Z2, Z3, and Z4, which are the 8 × 8 matrices (upper left, upper right, lower left, lower right) that our matrix can be split into.
>
> (d) For each of Z1, Z2, Z3, and Z4, determine the sum of the 64 elements of the mat

In [63]:

```
#a
Z = np.arange(256)
#b
Z = Z.reshape(16,16)
#c
Z1 = Z[0:8,0:8].copy()
Z2 = Z[0:8,8:].copy()
Z3 = Z[8:,0:8].copy()
Z4 = Z[8:,8:].copy()
#d
Z_list = [Z1,Z2,Z3,Z4]
Z_list_sum = [np.sum(i) for i in Z_list]

for i in range(len(Z_list_sum)):
    print()
    print(Z_list[i])
    print()
    print(f" Sum of Z{i+1} = {Z_list_sum[i]}")
```

```
[[  0   1   2   3   4   5   6   7]
 [ 16  17  18  19  20  21  22  23]
 [ 32  33  34  35  36  37  38  39]
 [ 48  49  50  51  52  53  54  55]
 [ 64  65  66  67  68  69  70  71]
 [ 80  81  82  83  84  85  86  87]
 [ 96  97  98  99 100 101 102 103]
 [112 113 114 115 116 117 118 119]]

 Sum of Z1 = 3808

[[  8   9  10  11  12  13  14  15]
 [ 24  25  26  27  28  29  30  31]
 [ 40  41  42  43  44  45  46  47]
 [ 56  57  58  59  60  61  62  63]
 [ 72  73  74  75  76  77  78  79]
 [ 88  89  90  91  92  93  94  95]
 [104 105 106 107 108 109 110 111]
 [120 121 122 123 124 125 126 127]]

 Sum of Z2 = 4320

[[128 129 130 131 132 133 134 135]
 [144 145 146 147 148 149 150 151]
 [160 161 162 163 164 165 166 167]
 [176 177 178 179 180 181 182 183]
```

```
 [192 193 194 195 196 197 198 199]
 [208 209 210 211 212 213 214 215]
 [224 225 226 227 228 229 230 231]
 [240 241 242 243 244 245 246 247]]

 Sum of Z3 = 12000

[[136 137 138 139 140 141 142 143]
 [152 153 154 155 156 157 158 159]
 [168 169 170 171 172 173 174 175]
 [184 185 186 187 188 189 190 191]
 [200 201 202 203 204 205 206 207]
 [216 217 218 219 220 221 222 223]
 [232 233 234 235 236 237 238 239]
 [248 249 250 251 252 253 254 255]]

 Sum of Z4 = 12512
```

Problem 2

In [64]:

```
A=np.array([[17, 24, 1, 8, 15],[23, 5, 7, 14, 16],[ 4, 6, 13, 20, 22],[10, 12, 19, 21, 3],[11, 18, 2
5, 2, 9]])
col_sums = [np.sum(c) for c in A.T]
row_sums = [np.sum(r) for r in A]
diagonal_sums = [np.trace(A),np.trace(np.fliplr(A))]
col_min = min(col_sums)
col_max = max(col_sums)
row_min = min(row_sums)
row_max = max(row_sums)
# test to validate results
test_results = row_min == row_max == col_min == col_max == diagonal_sums[0] == diagonal_sums[1]
print(f"Array is a Magic Square = {test_results}")
```

```
Array is a Magic Square = True
```

Problem 3

In [65]:

```
a = [2,3,5,2]
b = 45

#I recognize this runs in b^len(a) time but this was the only approach I could think of where broa
dcasting
# played a prominent role...
def solvefrob(coefs,b):
    j = np.array(list(itertools.combinations_with_replacement(range(0,b+1),len(coefs))))
    k = j * a  # mutiplication using broadcasting
    l = np.sum(k, axis=1) == b
    return j[l,:]

print("Solutions are: ")
print(solvefrob(a,b))
```

```
Solutions are:
[[ 0  0  1 20]
 [ 0  0  3 15]
 [ 0  0  5 10]
 [ 0  1  2 16]
 [ 0  1  4 11]
 [ 0  1  6  6]
 [ 0  2  3 12]
 [ 0  2  5  7]
 [ 0  3  4  8]
 [ 1  1  2 15]
 [ 1  1  4 10]
 [ 1  2  3 11]
 [ 1  2  5  6]
```

```
[ 1  2  3  6]
[ 1  3  4  7]
[ 2  2  3 10]
[ 2  2  5  5]
[ 2  3  4  6]
[ 3  3  4  5]]
```

Problem 4

Download the “College.csv” data set from the textbook’s website. You can find a description of the fields in the manual for the ISLR package.

> (a) Which schools are in the top 5 for the percentage of accepted students who actually enroll?
>
> (b) Which are the top 5 colleges in terms of total cost (out-of-state tuition, room and board, books, and personal spending)?
>
> (c) Which are the top 5 colleges in terms of profit per student? This should be based only on tuition revenue.

In [66]:

```
#Download the data, place into csv
df = pd.read_csv("http://faculty.marshall.usc.edu/gareth-james/ISL/College.csv").rename(columns={"
Unnamed: 0":"Name"})
# df.head(5)
```

In [67]:

```
#a
df_a = df.copy()
df_a['AcceptEnrollperc'] = df_a['Enroll']/df_a['Accept']
df_a['AEp_rank'] = df_a['AcceptEnrollperc'].rank(ascending=False)
df_a= df_a[df_a["AEp_rank"]<6].sort_values(by=['AEp_rank'])
print("Top 5 for the percentage of accepted students who actually enroll:")
for index,row in df_a.iterrows():
    print(f"{int(row['AEp_rank'])}. {row['Name']} {round(row['AcceptEnrollperc'],4)*100}%")
```

```
Top 5 for the percentage of accepted students who actually enroll:
1. California Lutheran University 100.0%
2. Brewton-Parker College 97.88%
3. Mississippi University for Women 93.83%
4. Peru State College 91.42%
5. Indiana Wesleyan University 86.52%
```

In [68]:

```
#b
df_b = df.copy()
columns_to_sum = ["Outstate","Room.Board","Books","Personal"]
df_b["total_cost"] = df_b[columns_to_sum].sum(axis=1)
df_b['cost_rank'] = df_b['total_cost'].rank(ascending=False)
df_b= df_b[df_b["cost_rank"]<6].sort_values(by=['total_cost'],ascending=False)
print("Top 5 colleges in terms of total cost (out-of-state tuition, room and board, books, and per
sonal spending):")
for index,row in df_b.iterrows():
    print(f"{int(row['cost_rank'])}. {row['Name']} ${row['total_cost']}")
```

```
Top 5 colleges in terms of total cost (out-of-state tuition, room and board, books, and personal s
pending):
1. Yale University $29095
2. Massachusetts Institute of Technology $28400
3. Princeton University $28060
4. Georgetown University $27801
5. Brandeis University $27540
```

In [69]:

```
#c
```

```
df_c = df.copy()
df_c.eval('profit_per_stud = Outstate - Expend',inplace=True)
df_c['profit_rank'] = df_c['profit_per_stud'].rank(ascending=False)
df_c= df_c[df_c["profit_rank"]<6].sort_values(by=['profit_rank'])
print("Top 5 colleges in terms of profit per student? This should be based only on tuition revenue
:")
for index,row in df_c.iterrows():
    print(f"{int(row['profit_rank'])}. {row['Name']} steals ${row['profit_per_stud']} from each
student")
```

```
Top 5 colleges in terms of profit per student? This should be based only on tuition revenue:
1. Elmira College steals $7488 from each student
2. Buena Vista College steals $6973 from each student
3. Franklin Pierce College steals $6902 from each student
4. Providence College steals $6265 from each student
5. Saint Anselm College steals $6231 from each student
```

Problem 5

Download the “Credit.csv” data set from the textbook’s website.

> (a) Create a new column for the credit utilization for each person (balance divided by limit)
>
> (b) What is the average income level for each education level?
>
> (c) What is the average income for each age block (Under 19, 20-29, 30-39, etc)?
>
> (d) Create scatter plots of the following. State an opinion you can infer from the scatter plot, if there is one.
>
> > i. Credit Limit vs Rating
> >
> > ii. Credit Limit vs Income
> >
> > iii. Balance vs Income
> >
> > iv. Income vs Education
> >
> > v. Limit vs Education

In [70]:

```
df = pd.read_csv("http://faculty.marshall.usc.edu/gareth-james/ISL/Credit.csv")
df.columns = map(str.lower,df.columns)

#Getting the binning out of the way so the answers are less cluttered
age_bins = [0,20,30,40,50,60,70,80,90,100,10000]
age_label = ["0-19","20-30","30-40","40-50","50-60","60-70","70-80","80-90","90-100","100+"]
df['age_group'] = pd.cut(df['age'],age_bins, labels=age_label)
df.head(2)
```

Out[70]:

	unnamed: 0	income	limit	rating	cards	age	education	gender	student	married	ethnicity	balance	age_group
0	1	14.891	3606	283	2	34	11	Male	No	Yes	Caucasian	333	30-40
1	2	106.025	6645	483	3	82	15	Female	Yes	Yes	Asian	903	80-90

In [71]:

```
#a)
df.eval('credit_util = balance / limit', inplace=True)
df.head(3)
```

Out[71]:

	unnamed: 0	income	limit	rating	cards	age	education	gender	student	married	ethnicity	balance	age_group	credit_util

	unnamed: 0	income	limit	rating	cards	age	education	gender	student	married	ethnicity	balance	age_group	credit_util
0	1	14.891	3606	283	2	34	11	Male	No	Yes	Caucasian	333	30-40	0.092348
1	2	106.025	6645	483	3	82	15	Female	Yes	Yes	Asian	903	80-90	0.135892
2	3	104.593	7075	514	4	71	11	Male	No	No	Asian	580	70-80	0.081979

In [72]:

```
#b)
df_b = df.copy()
df_b = df_b.groupby(['education'], as_index=False)['income'].mean()
df_b.columns = ['education_level','avg_income']
df_b
```

Out[72]:

	education_level	avg_income
0	5	34.142000
1	6	55.376800
2	7	45.153625
3	8	66.939500
4	9	40.387600
5	10	35.514208
6	11	53.092606
7	12	43.401405
8	13	40.077868
9	14	46.467542
10	15	43.938000
11	16	42.881640
12	17	55.284235
13	18	38.112364
14	19	41.441100
15	20	37.607500

In [73]:

```
#c)
df_c = df.copy()
df_c = df_c.groupby(['age_group'],as_index=False)['income'].mean()
df_c['income'].fillna("no records", inplace = True)
df_c.columns = ['age_group','avg_income']
df_c
```

Out[73]:

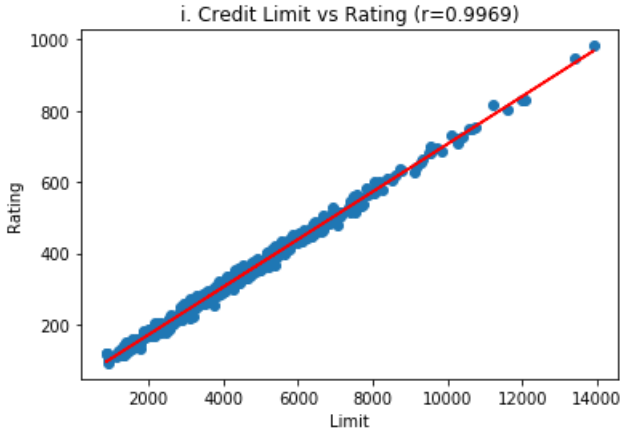
	age_group	avg_income
0	0-19	no records
1	20-30	31.4709
2	30-40	40.4161
3	40-50	49.1126
4	50-60	39.1194
5	60-70	45.1003
6	70-80	41.8268
7	80-90	69.3252
8	90-100	167.337
9	100+	no records

d-i

In [74]:

```
#d-i
plt.scatter(df['limit'], df['rating'])
plt.title('i. Credit Limit vs Rating')
plt.ylabel("Rating")
plt.xlabel("Limit")

slope, intercept, r_value, p_value, std_err = stats.linregress(df['limit'], df['rating'])
plt.plot(df['limit'], intercept + slope * df['limit'], '-',c="red")
plt.title(f'i. Credit Limit vs Rating (r={round(r_value,4)})')
plt.show()
```



The trend here is relatively obvious, there is a nearly a perfect postive correlation (r=0.9969) between credit limit and credit rating and credit limit. This relationship is likely due to the fact that calculate to determine credit rating takes into account credit limit.

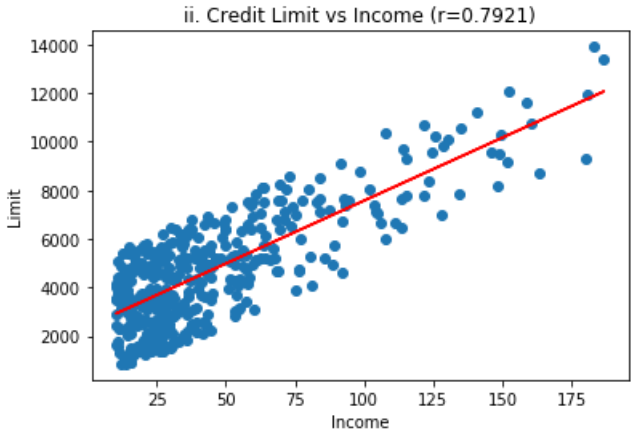
d-ii

In [75]:

```
#d-ii
plt.scatter(df['income'], df['limit'])

plt.xlabel("Income")
plt.ylabel("Limit")

slope, intercept, r_value, p_value, std_err = stats.linregress(df['income'], df['limit'])
plt.plot(df['income'], intercept + slope * df['income'], '-',c="red")
plt.title(f'ii. Credit Limit vs Income (r={round(r_value,4)})')
plt.show()
```



Note that I have adjusted the X-Axis to reflect income level and the Y-Axis to reflect credit limit, this has been done because income is independent of credit limit, but credit limit is not typically independent of income. The trend here is relatively obvious, there is a strong postive correlation (r=0.7921) between income and credit limit. This means that as person's income increases so does thier

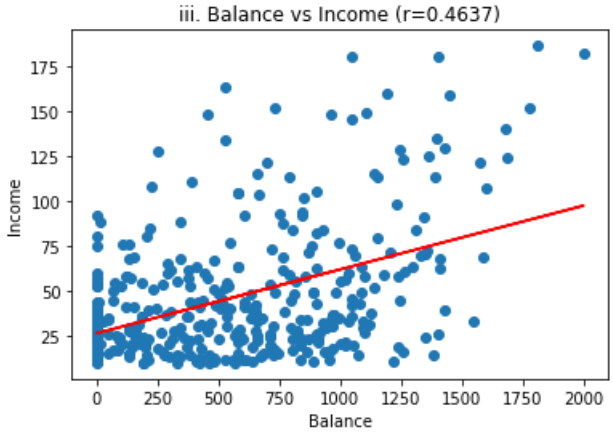
strong postive correlation (r=0.7921) between income and credit limit. This means that as person's income increases so does thier credit limit.

d-iii

In [76]:

```
#d-iii
plt.scatter(df['balance'], df['income'])
plt.ylabel("Income")
plt.xlabel("Balance")

slope, intercept, r_value, p_value, std_err = stats.linregress(df['balance'], df['income'])
plt.plot(df['balance'], intercept + slope * df['balance'], '-',c="red")
plt.title(f'iii. Balance vs Income (r={round(r_value,4)})')
plt.show()
```



There is a weak correlation between balance and income with r=0.4637. Interestingly enough, as income level increases beyond 100 the correlation does appear to become stronger and more predictive, though not as strong as our previous scatter plots.

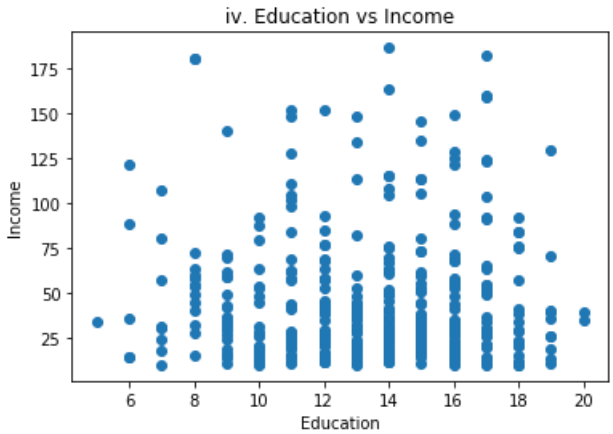
d-iv

In [77]:

```
#d-iv
plt.scatter(df['education'], df['income'])
plt.title('iv. Education vs Income')
plt.xlabel("Education")
plt.ylabel("Income")
```

Out[77]:

```
Text(0, 0.5, 'Income')
```



There is no obvious relationship between education level and income based on this chart, this observation is seemingly against

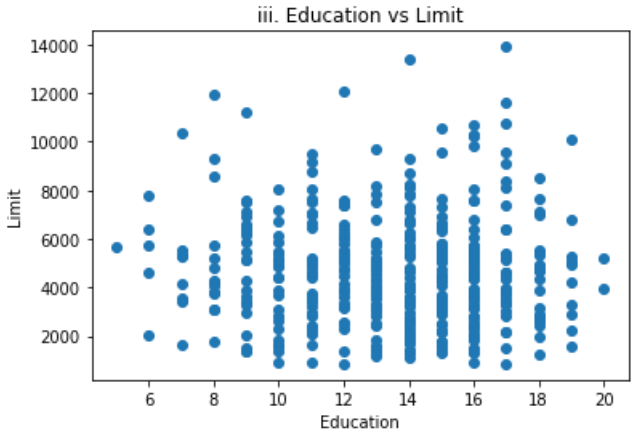
expectation as we would expect income levels to increase with education attainment. Based on the scatter plot above, we see that generally there tends to be a concentration of people at the lower potions of the income spectrum with the majority falling below 75, though this estimate is an eyeball estimate.

In [78]:

```
#d-v
plt.scatter(df['education'], df['limit'])
plt.title('iii. Education vs Limit')
plt.xlabel("Education")
plt.ylabel("Limit")
```

Out[78]:

```
Text(0, 0.5, 'Limit')
```



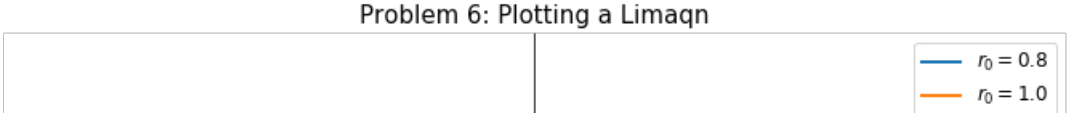
As with **d-iv)** there appears to be no obvious relationship between education level and credit limit based on this chart, this observation contradict expectation as we would expect credit limit to increase with educational attainment, though this assumption is largely based on the assumption that income level and educational attaintment are postively correlated.

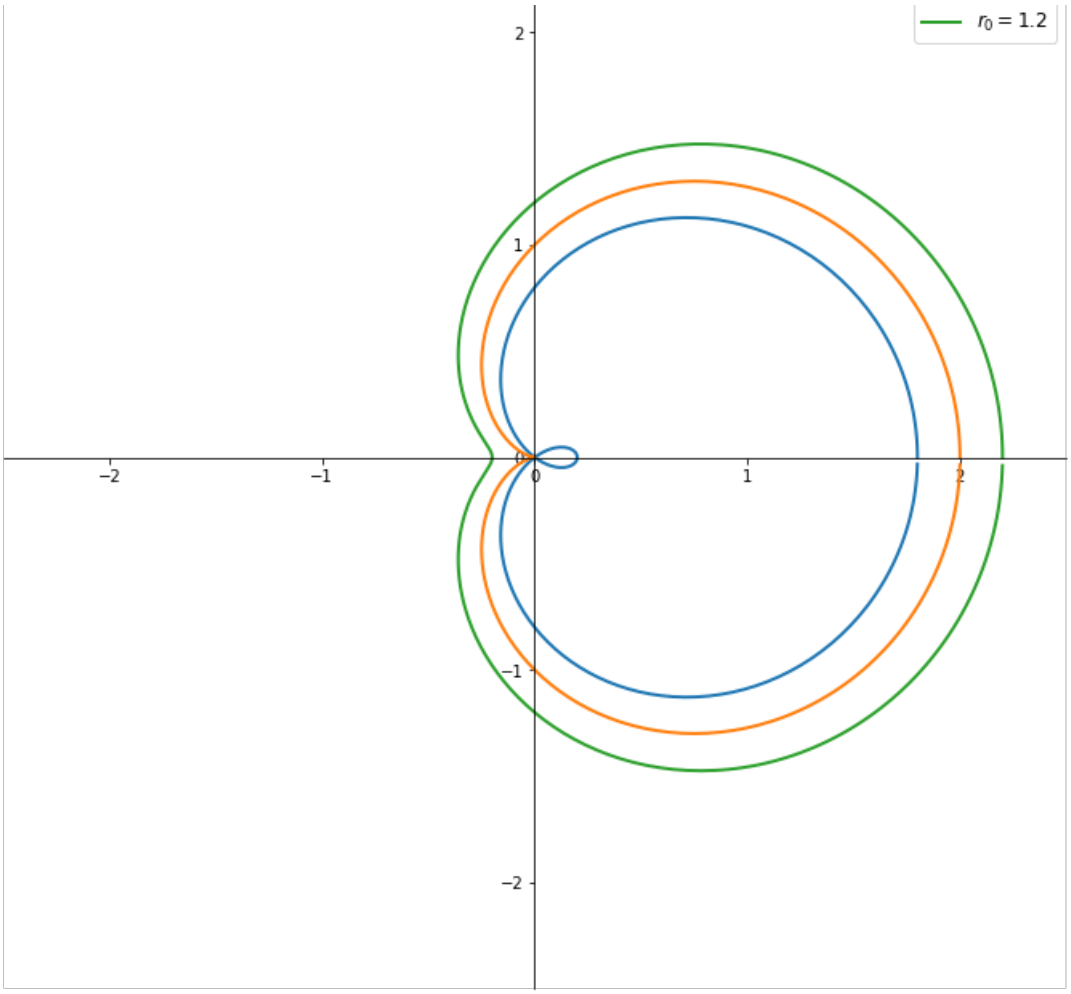
Based on the scatter plot above, we see that generally there is a concentration of people at the lower portions of the income spectrum with the majority falling below 9000, though this estimate is an eyeball estimate.

Problem 6

In [79]:

```
r_0_array = [0.8,1.0,1.2]
fig = plt.figure(figsize=(10,10),linewidth=10)
ax = fig.add_subplot(111)
plt.title(r'Problem 6: Plotting a Limaqn',fontsize=15)
for r_0 in r_0_array:
    r_array = [r_0+np.cos(np.deg2rad(i)) for i in range(0,360) ]
    x_array = [r_array[i]*np.cos(np.deg2rad(i)) for i in range(0,360)]
    y_array = [r_array[i]*np.sin(np.deg2rad(i)) for i in range(0,360)]
    ax.plot(x_array,y_array,linewidth=2)
plt.legend([r'$r_0 = 0.8$',r'$r_0 = 1.0$',r'$r_0 = 1.2$'],fontsize=12)
plt.tight_layout()
ax.set_ylim([-2.5,2.5])
ax.set_xlim([-2.5,2.5])
ax.spines['left'].set_position('zero')
ax.spines['right'].set_color('none')
ax.spines['bottom'].set_position('zero')
ax.spines['top'].set_color('none')
ax.xaxis.set_ticks_position('bottom')
ax.yaxis.set_ticks_position('left')
plt.savefig("./Problem_6.png")
plt.show()
```




$r_0 = 1.2$
2
1
0
−1
−2
−2
−1
0
1
2