**LAB RECORD**

**2018 – 2019**

Mathematical Models Using Python Programming

MAT 451

**Name:** Jeevan Koshy **Reg. No:** 1740256

**TABLE OF CONTENTS**

**Sl No. Name Date Pg No**

|  |  |  |  |
| --- | --- | --- | --- |
| 2 | Inverse Determinant & Eigen Values | 15th November 2018 | 3 – 4 |
| 3 | Transpose & Upper/Lower Triangular Parts | 17th November 2018 | 5 – 6 |
| 4 | Solving Linear Systems | 22nd November 2018 | 6 – 10 |
| 5 | Plotting of scalar & vector fields | 24th November 2018 | 10 – 13 |
| 6 | Mathematical model: interest rates | 10th January 2019 | 14 |
| 7 | Mathematical model: Growth of population/Exponential model | 12th January 2019 | 15 – 16 |
| 8 | Mathematical model: Logistic Growth | 24th January 2019 | 17 - 19 |

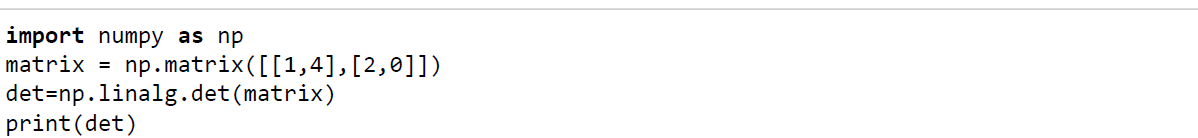
**Lab 2**

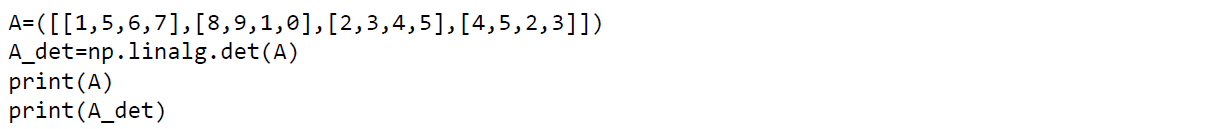
**Topic:** Inverse, Determinant & Eigen Values

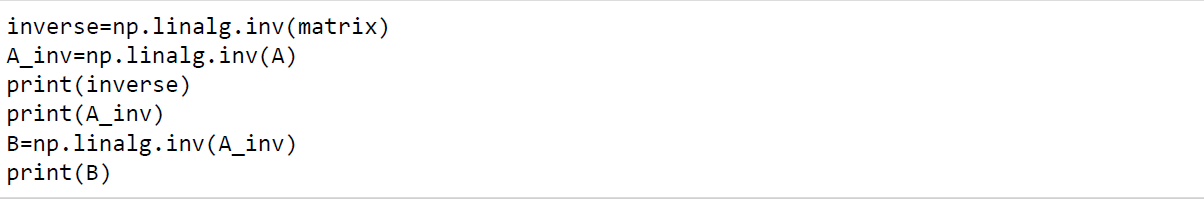
**Date:** 15th November 2018

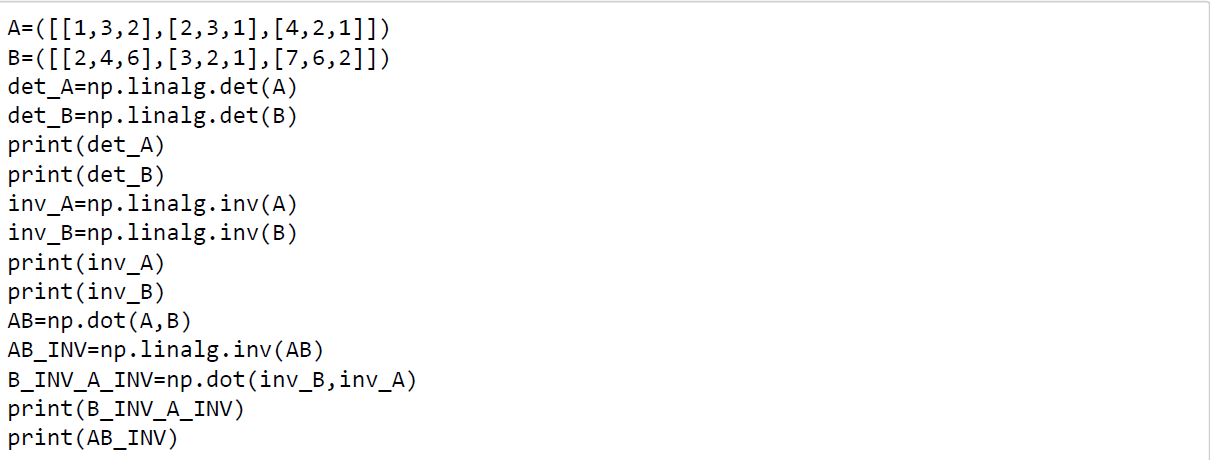
**Aim:** To find the determinant, inverse and eigen values of matrices.

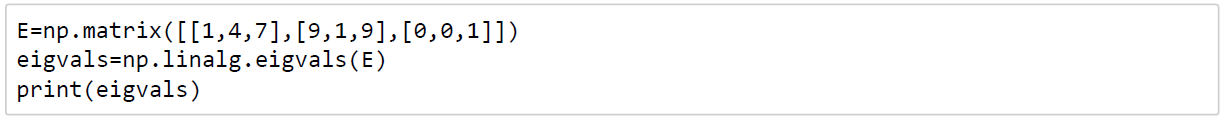
**Source Code:**











**Output (Graphs/Tables):**

-8.0

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

-86.0

[[ 0. 0.5 ]

[ 0.25 -0.125]]

[[-0.39534884 0.09302326 0.65116279 -0.1627907 ]

[ 0.34883721 -0.02325581 -0.6627907 0.29069767]

[ 0.02325581 0.46511628 0.75581395 -1.31395349]

[-0.06976744 -0.39534884 -0.26744186 0.94186047]]

[[ 1.00000000e+00 5.00000000e+00 6.00000000e+00 7.00000000e+00]

[ 8.00000000e+00 9.00000000e+00 1.00000000e+00 2.35922393e-16]

[ 2.00000000e+00 3.00000000e+00 4.00000000e+00 5.00000000e+00]

[ 4.00000000e+00 5.00000000e+00 2.00000000e+00 3.00000000e+00]]

-9.0

24.0

[[-0.11111111 -0.11111111 0.33333333]

[-0.22222222 0.77777778 -0.33333333]

[ 0.88888889 -1.11111111 0.33333333]]

[[-0.08333333 1.16666667 -0.33333333]

[ 0.04166667 -1.58333333 0.66666667]

[ 0.16666667 0.66666667 -0.33333333]]

[[-0.5462963 1.28703704 -0.52777778]

[ 0.93981481 -1.97685185 0.76388889]

[-0.46296296 0.87037037 -0.27777778]]

[[-0.5462963 1.28703704 -0.52777778]

[ 0.93981481 -1.97685185 0.76388889]

[-0.46296296 0.87037037 -0.27777778]]

[ 7. -5. 1.]

**Conclusion:**

From the above output, we have calculated the determinants of a matrix as well as found it’s eigen values with it’s inverse. Using these codes, we can find for any matrix – it’s determinant, inverse & eigen value.

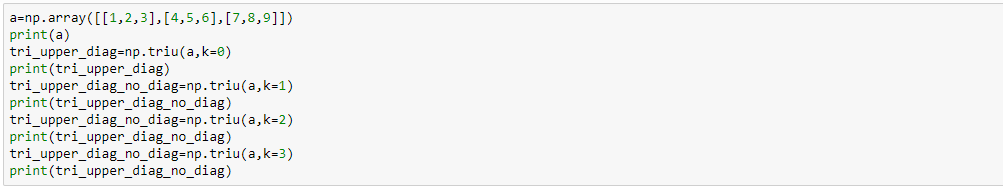
**Lab 3**

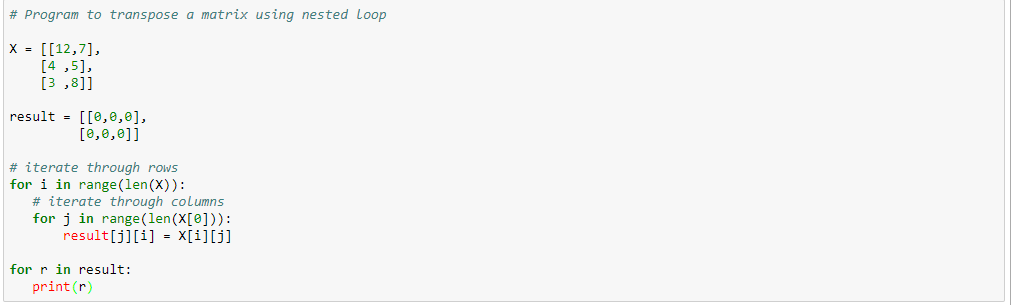
**Topic:** Transpose & Upper/Lower Triangular Parts

**Date:** 17th November 2018

**Aim:** To find the transpose, upper & lower triangular parts of a matrix.

**Source Code:**





**Output (Graphs/Tables):**

[[1 2 3]

[4 5 6]

[7 8 9]]

[[1 2 3]

[0 5 6]

[0 0 9]]

[[0 2 3]

[0 0 6]

[0 0 0]]

[[0 0 3]

[0 0 0]

[0 0 0]]

[[0 0 0]

[0 0 0]

[0 0 0]]

[12, 4, 3]

[7, 5, 8]

**Conclusion:**

From the above codes, we have found the transpose, upper & lower triangular parts of a matrix.

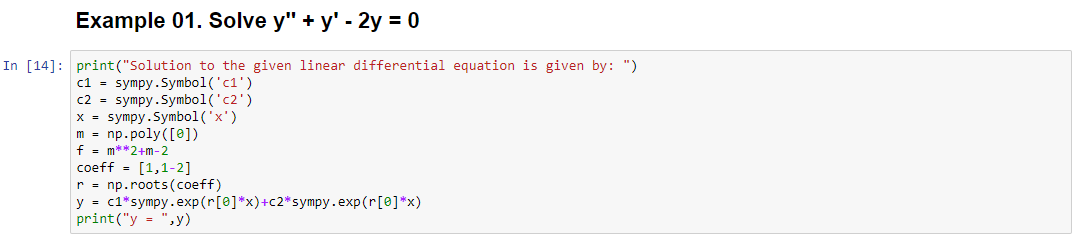
**Lab 4**

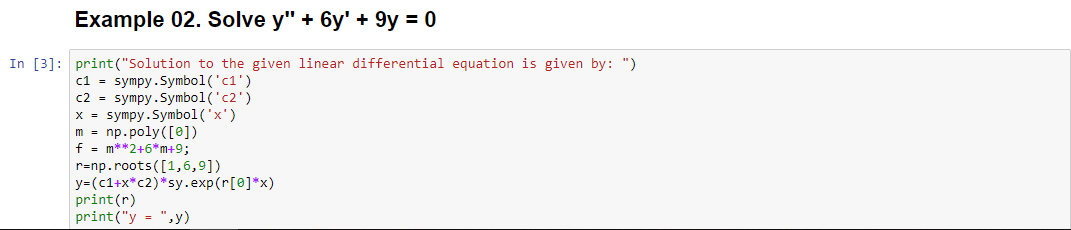
**Topic:** Solving Linear Systems

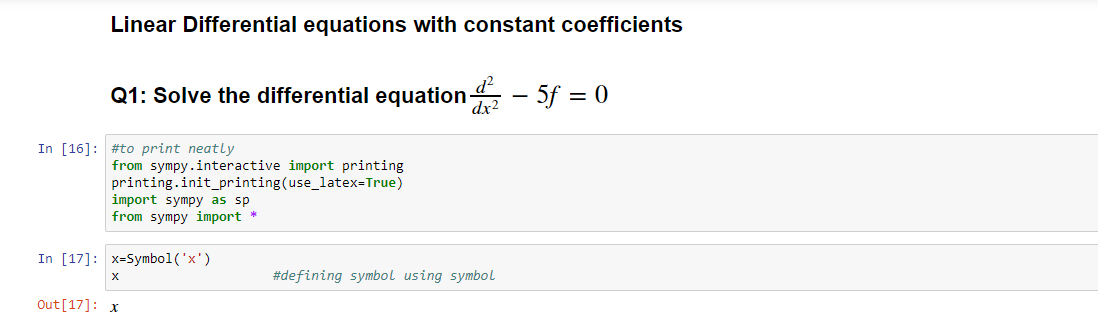
**Date:** 22nd November 2018

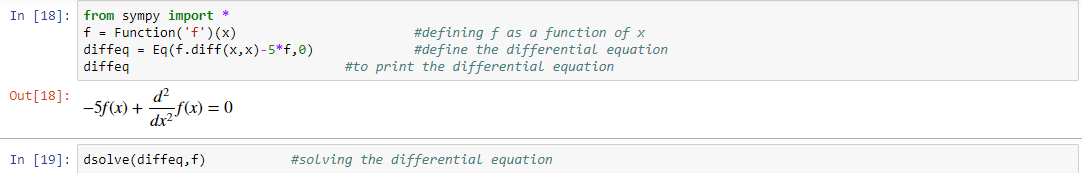
**Aim:** To solve linear systems in Python

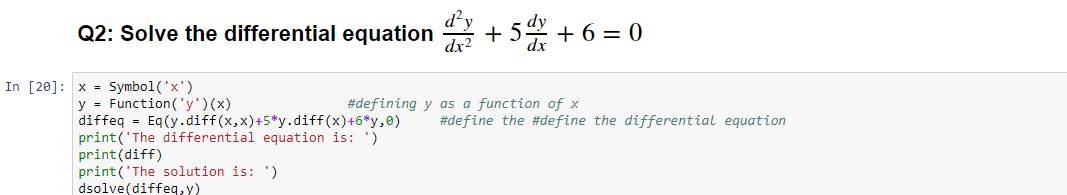
**Source Code:**

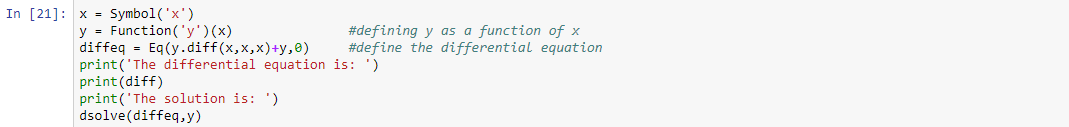


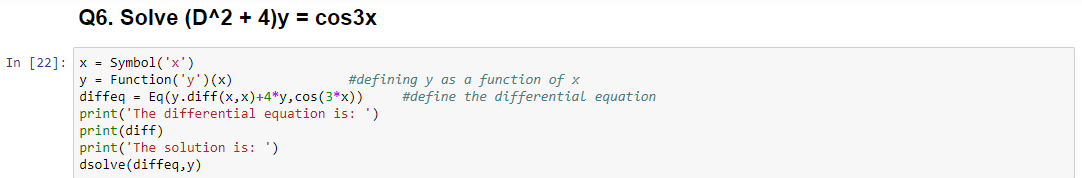


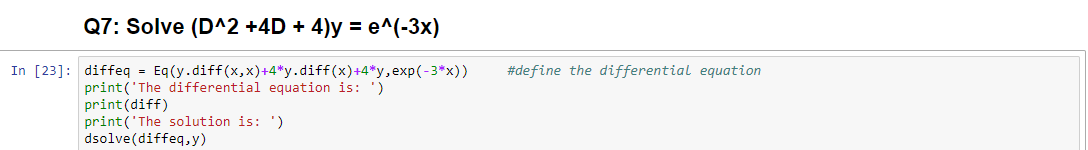


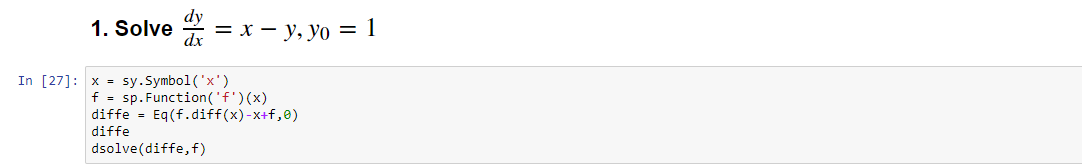


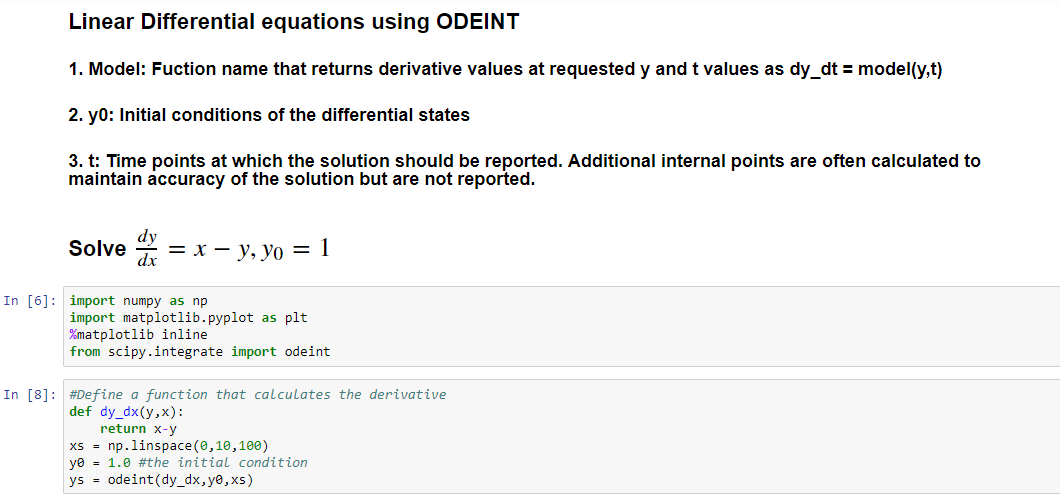


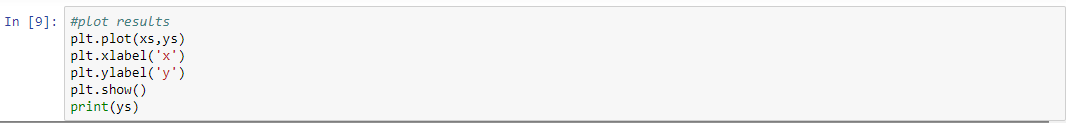


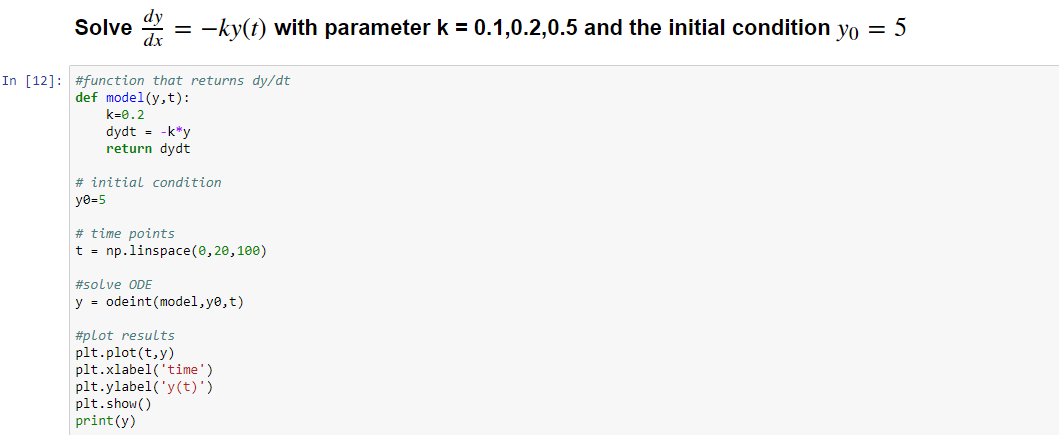


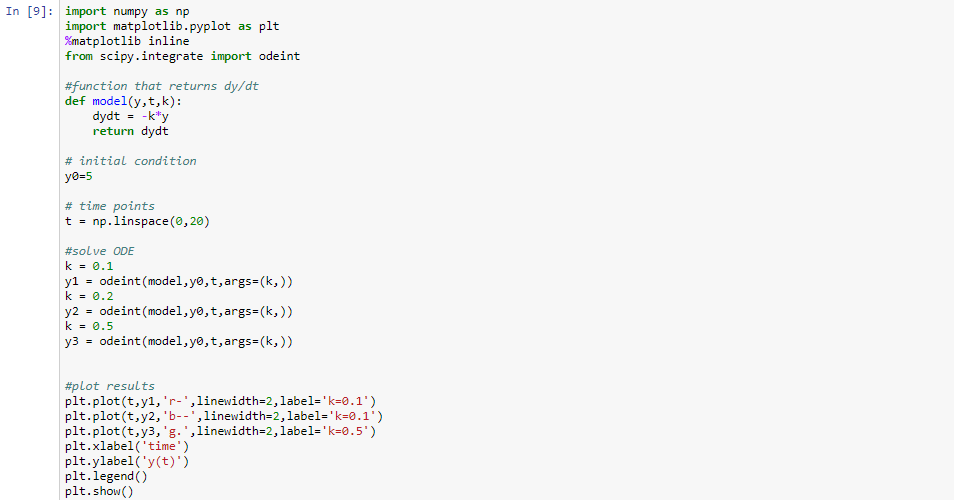














**Output (Graphs/Tables):**

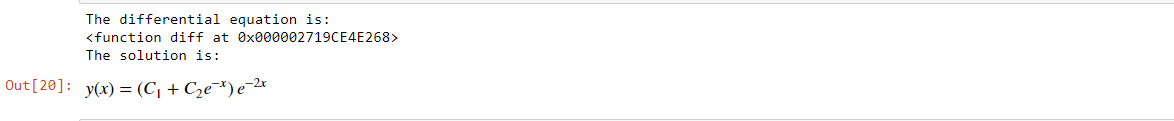
Solution to the given linear differential equation is given by:

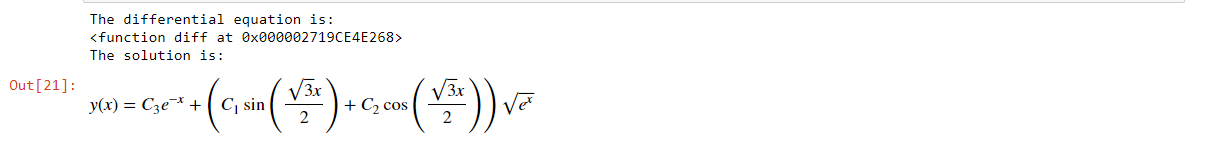
y = c1\*exp(1.0\*x) + c2\*exp(1.0\*x)

Solution to the given linear differential equation is given by:

[-3. +3.72529030e-08j -3. -3.72529030e-08j]

y = (c1 + c2\*x)\*exp(x\*(-3.0 + 3.72529029846191e-8\*I))

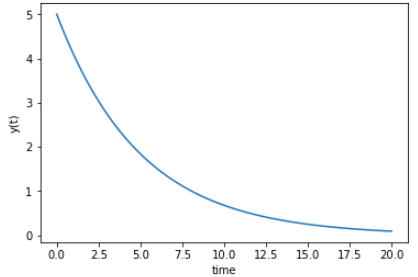
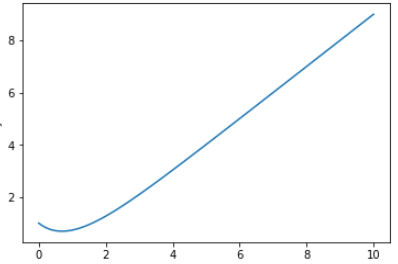


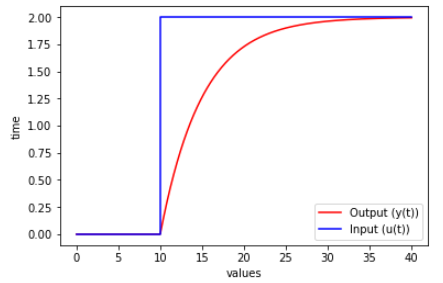
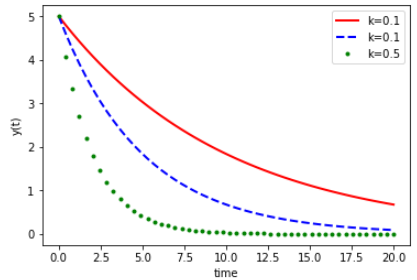












**Conclusion:**

With the above codes & outputs, we have solved and found linear systems in Python. A few graphs have also been plotted for the different linear differential equations.

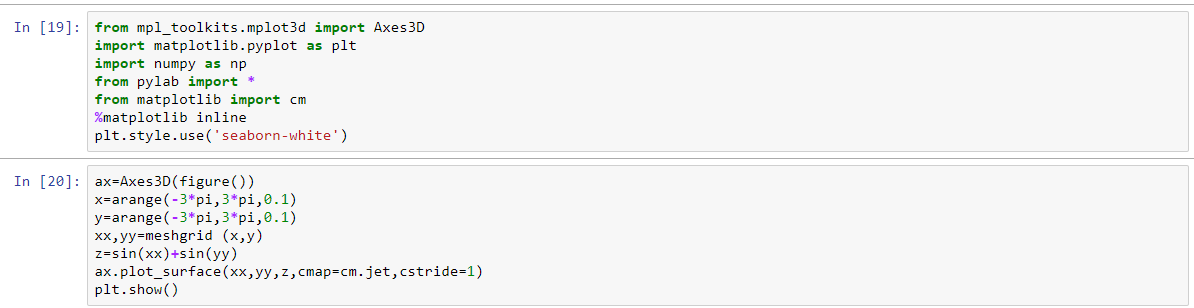
**Lab 5**

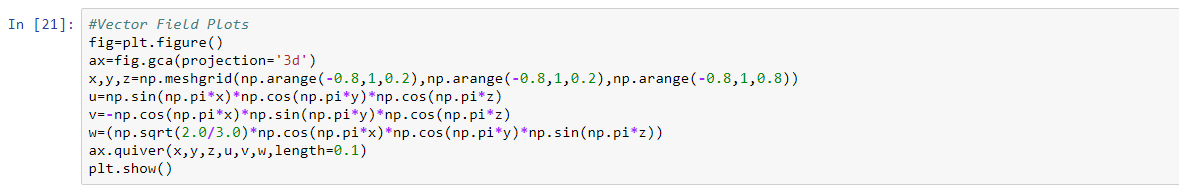
**Topic:** Plotting of scalar & vector fields

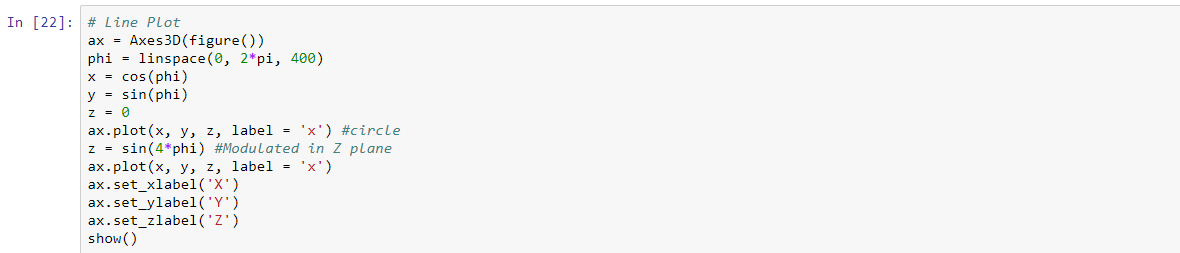
**Date:** 24th November 2018

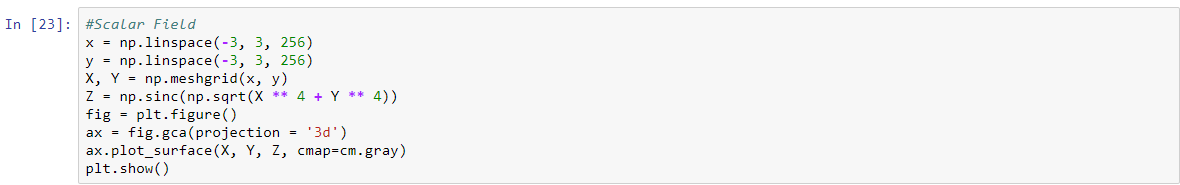
**Aim:** To plot scalar & vector fields using Python and to find the cross product of vectors

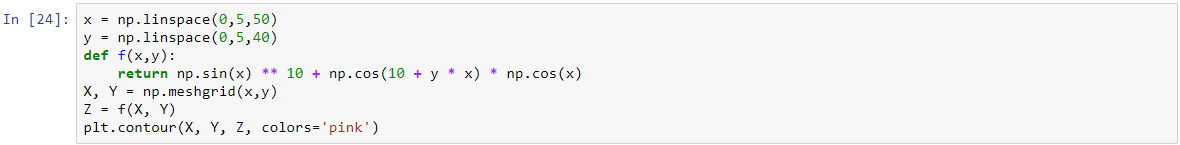
**Source Code:**

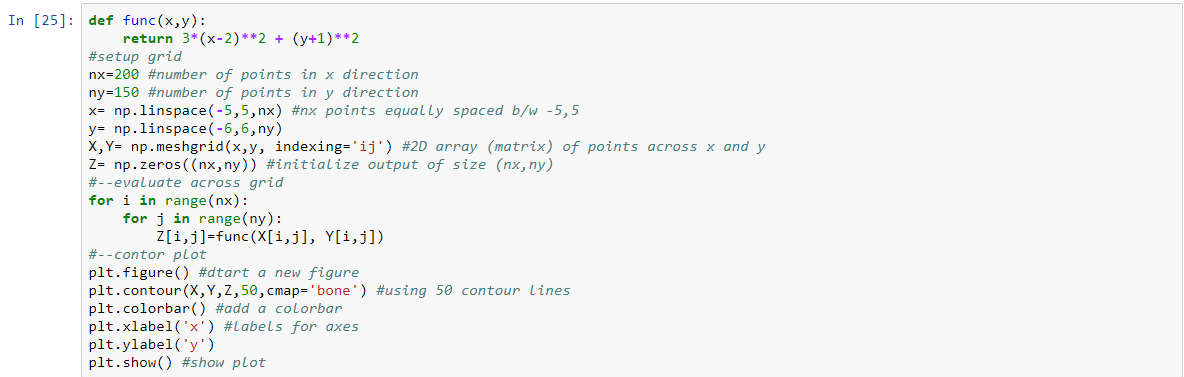




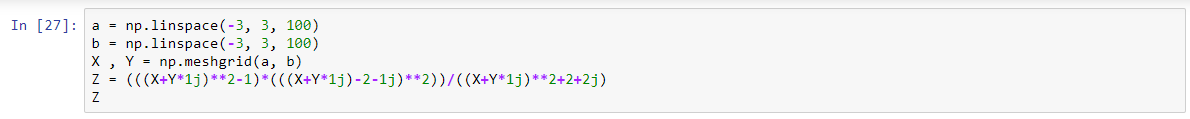


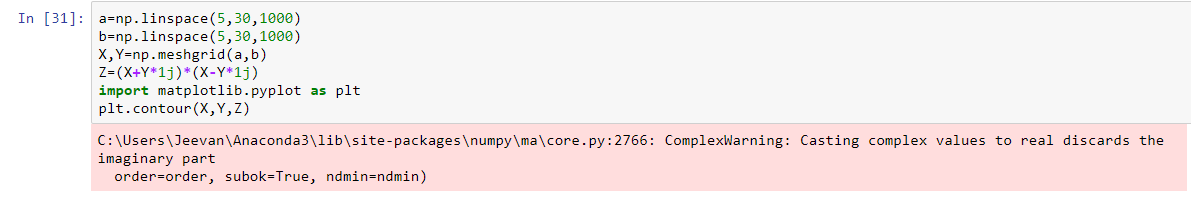


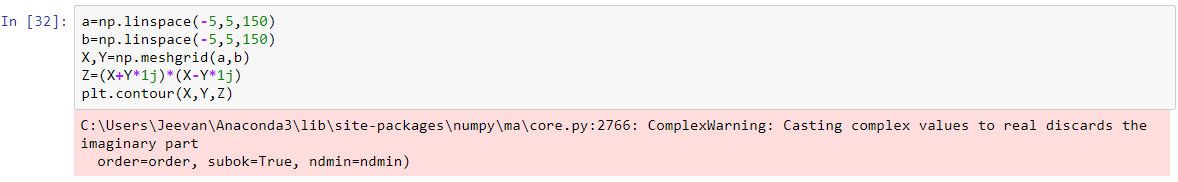




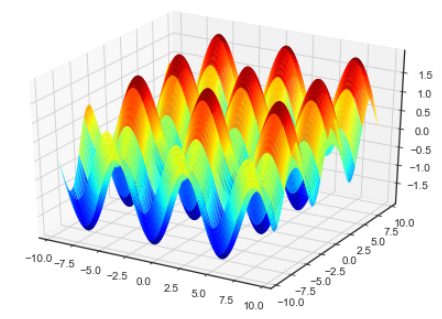
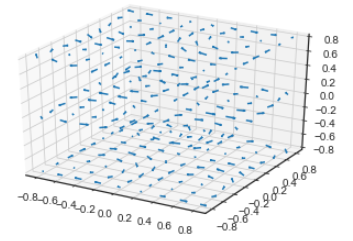


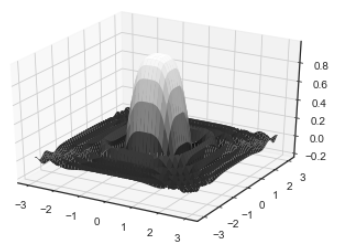
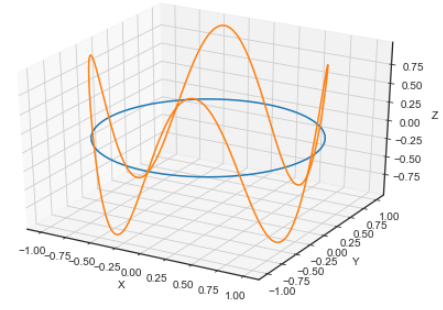


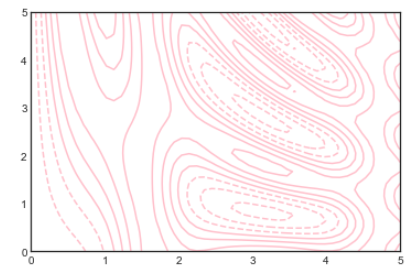
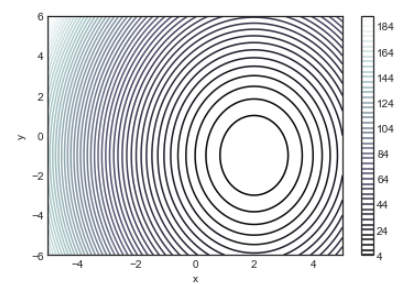


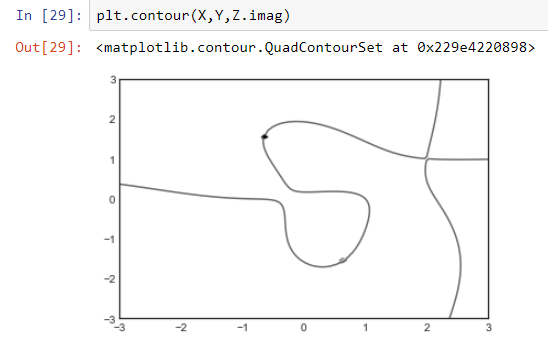
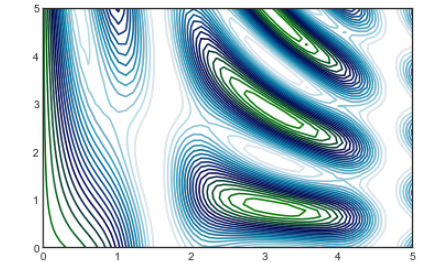


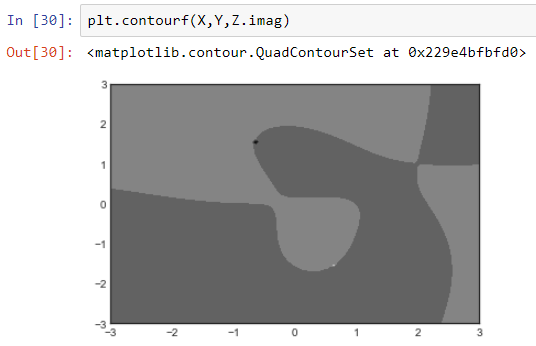
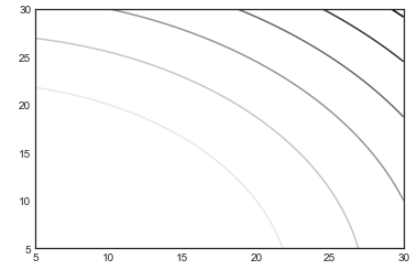
**Output (Graphs/Tables):**

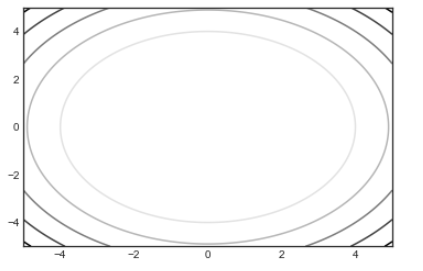
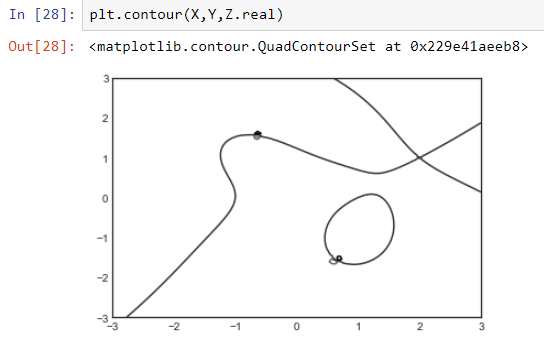
 







**Conclusion:**

From the above graphs, we can understand how scalar and vector fields are plotted with the codes entered.

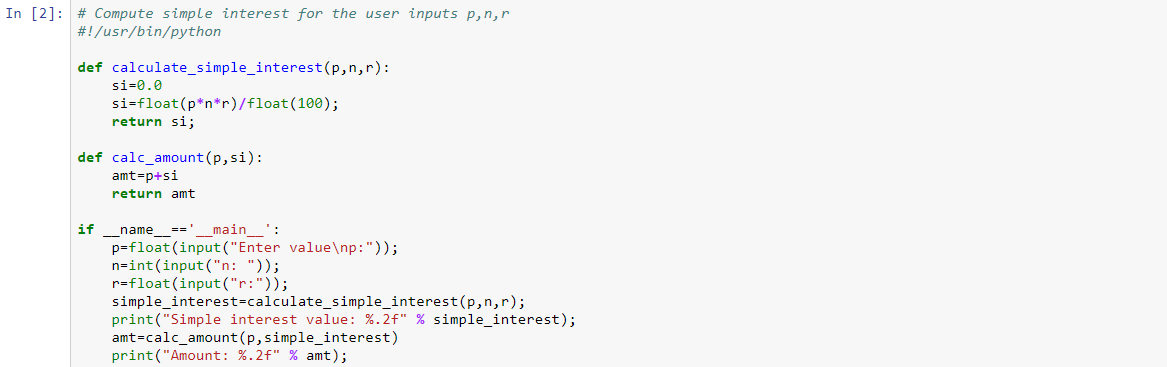
**Lab 6**

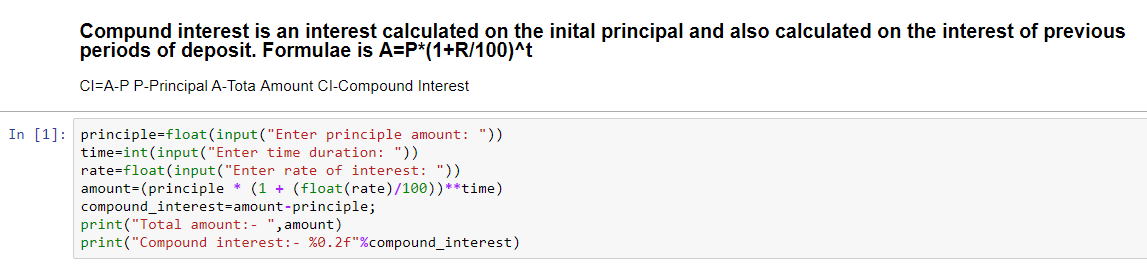
**Topic:** Mathematical model: Interest rates

**Date:** 10th January 2019

**Aim:**

**Source Code:**





**Output (Graphs/Tables):**

Enter value

p:100

n: 10

r:5.0

Simple interest value: 50.00

Amount: 150.00

Enter principle amount: 100

Enter time duration: 10

Enter rate of interest: 12

Total amount:- 310.5848208344212

Compound interest:- 210.58

**Conclusion:**This is how interest rates are calculated in Python.

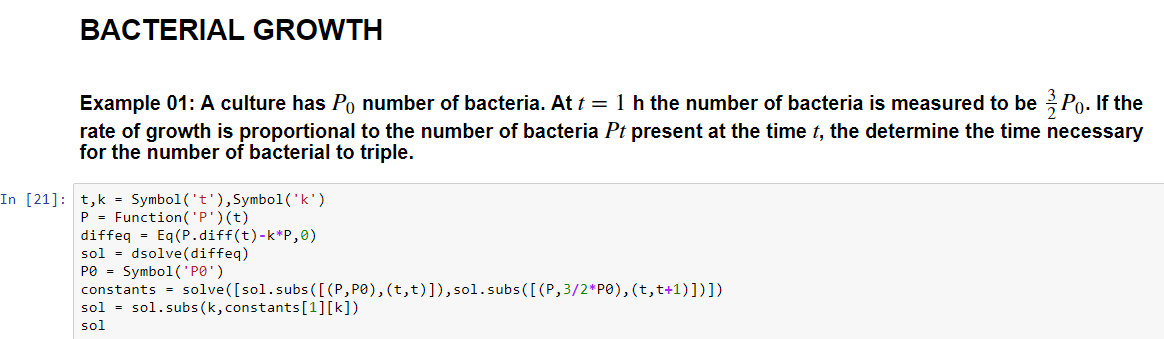
**Lab 7**

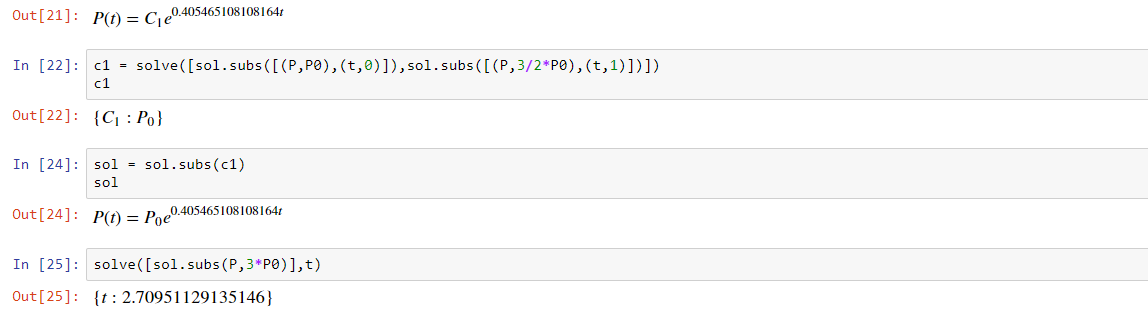
**Topic:** Mathematical model: Growth of population/Exponential model

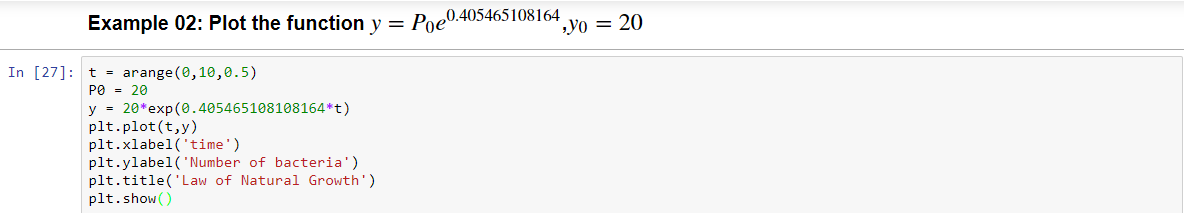
**Date:**  12th January 2019

**Aim:** To find the bacterial growth of a population

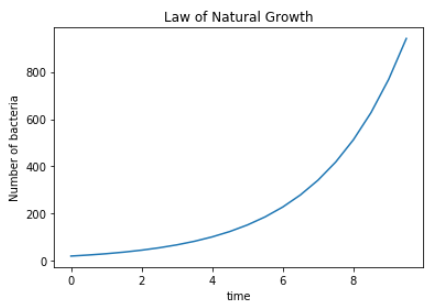
**Source Code:**







**Output (Graphs/Tables):**



**Conclusion:**

The law of natural growth has been plotted with help of a few python codes. There is also an example or two have been used to find the time for bacteria to grow.

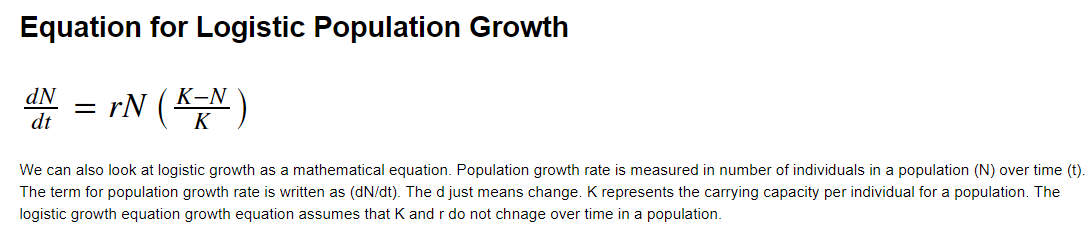
**Lab 8**

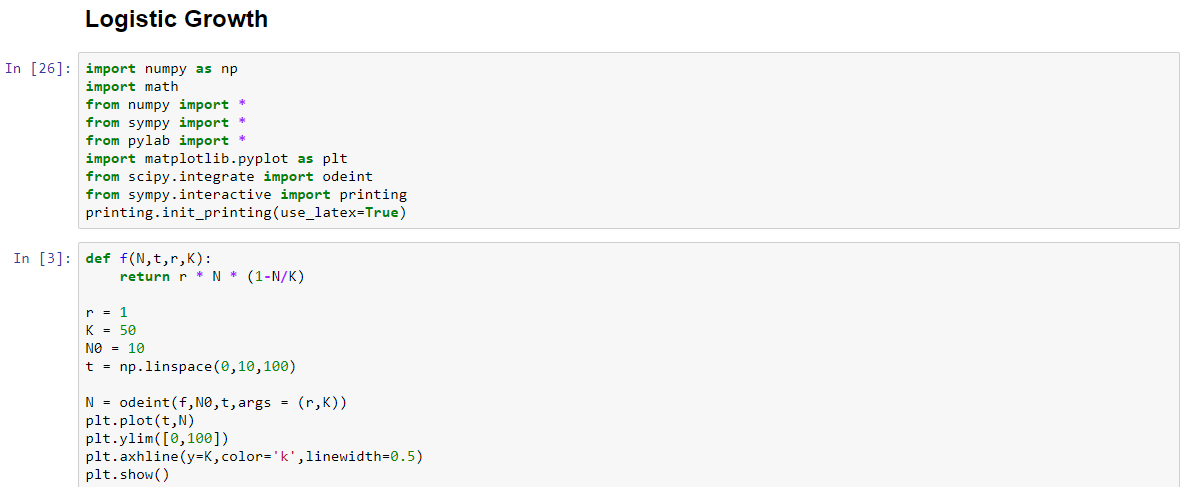
**Topic:** Mathematical model: Logistic Growth

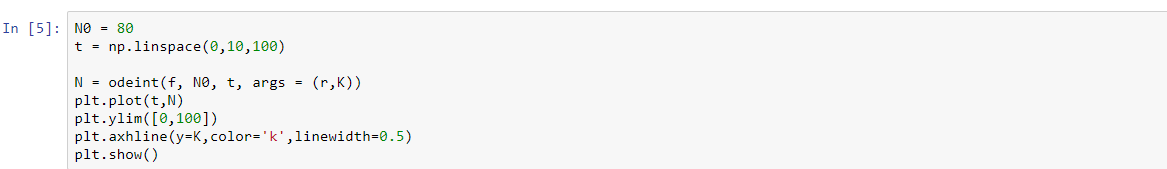
**Date:** 24th January 2019

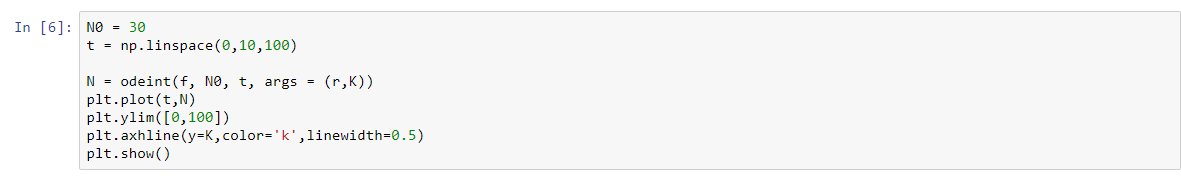
**Aim:** To build a mathematical model finding the logistic growth in a population

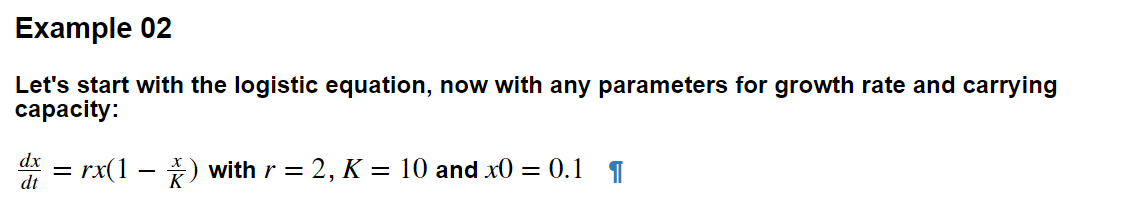
**Source Code:**

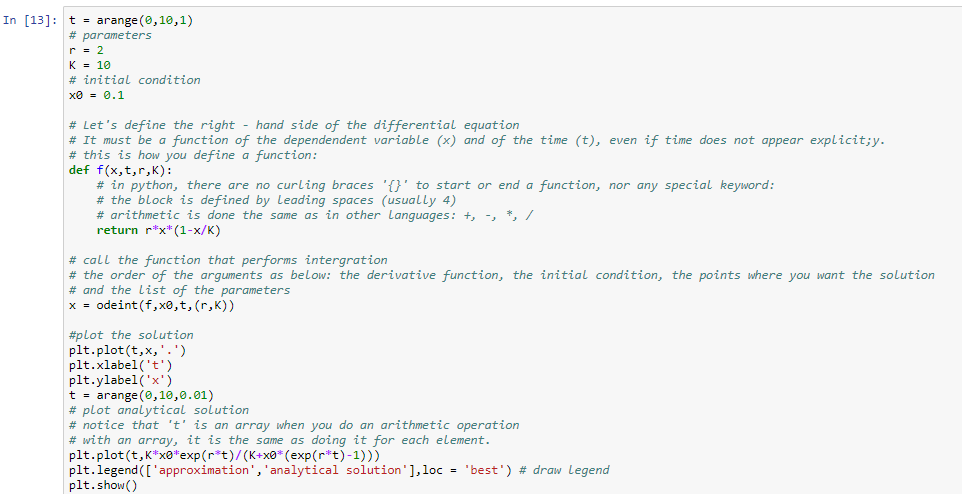




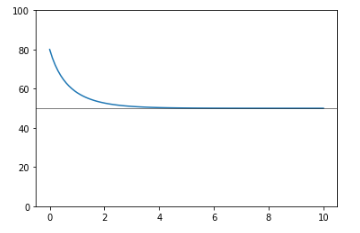
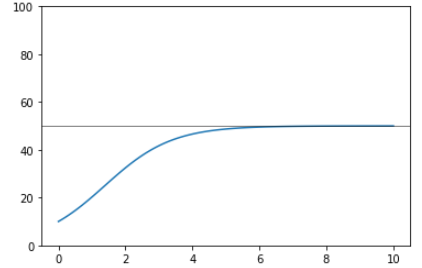


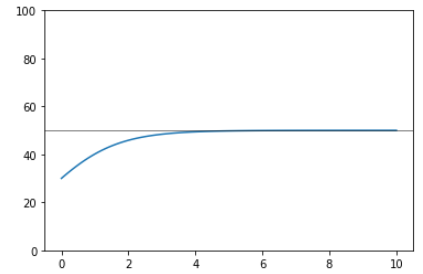
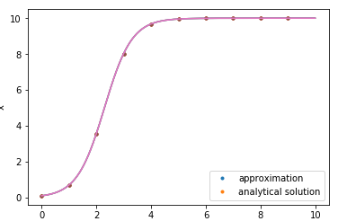






**Output (Graphs/Tables):**



**Conclusion:**

From the above graph, the logistic growth of a population has been calculated and a few graphs have also been plotted in Python with the help of some code to demonstrate the growth.

-------------------------------------------------------------------------