University of Canberra

Faculty of Science and Technology

**Programming for Data Science G (11521)**

**Week 4 Tutorial**

**Modules – Input/Output – Exception Handling – Assignment 1**

**Objectives**

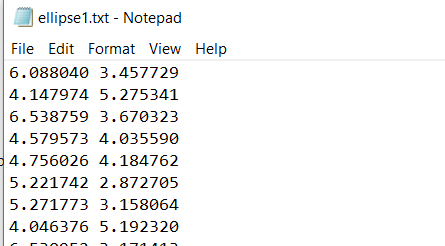
* To create a module and add functions for input/output to this module
* To implement functions for reading data and calculating distances

**Use Visual Studio or PyCharm or Spyder (in Anaconda) to Create a New Python Project**

* Use your own computer or login to UC Student Virtual Desktop <https://frame.nutanix.com/university-of-canberra/ditm/uc-remote-access-student/launchpad/uc-virtual-desktop-student>
* Open Visual Studio or PyCharm or Spyder (in Anaconda) to create a new project and name it **Week4Tutorial**.
* Add a new Python file to this project and name it **io\_data\_module.py**.
* Your project now has 2 Python files: **Week4Tutorial.py** file (for the main program) and **io\_data\_module.py** (the module file for input/output methods)

**Add function to read data from file to io\_data\_module.py**

* **Function to read 2-dimensional data from file and save data to tuples**. The dataset is in a text file below. Each line has 2 values which are x & y coordinates of a data sample.



Copy the following function to **io\_data\_module.py**

#Function to read 2D data from file and save data to a list of tuples

def **read\_data\_file**(filename):

dataset = [] #dataset is a python list

f = None

try:

f = open(filename, 'r')

while True:

line = f.readline()

if len(line) == 0: #end of file

break

line = line.replace('\n', '') #remove end of line \n character

xystring = line.split(' ') #x y coordinates in string format

#use split function to separate x & y strings then

#use float function to convert x & y strings to x & y numbers and

#add them as a tuple (x, y) to dataset that is a list

dataset.append((float(xystring[0]), float(xystring[1])))

except Exception as ex:

print(ex.args)

finally:

if f:

f.close()

return dataset

#end of function

**Call the function in Week4Tutorial.py**

* Call the function in the program (**Week4Tutorial.py**) as follows

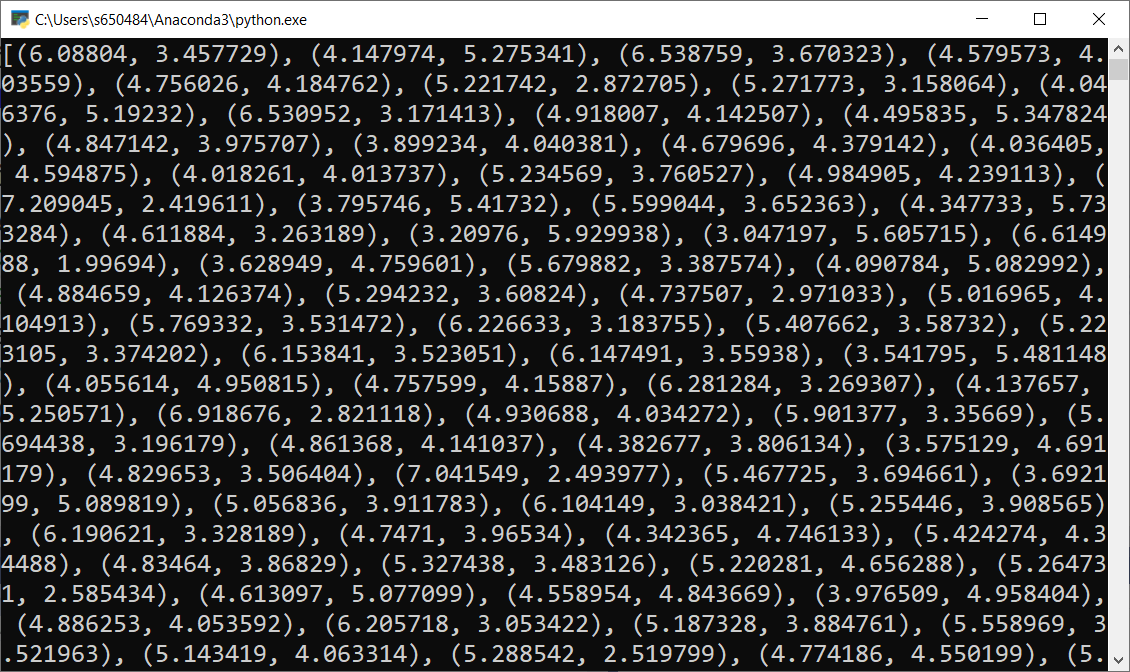
import io\_data\_module as iodata

#Open file and read data

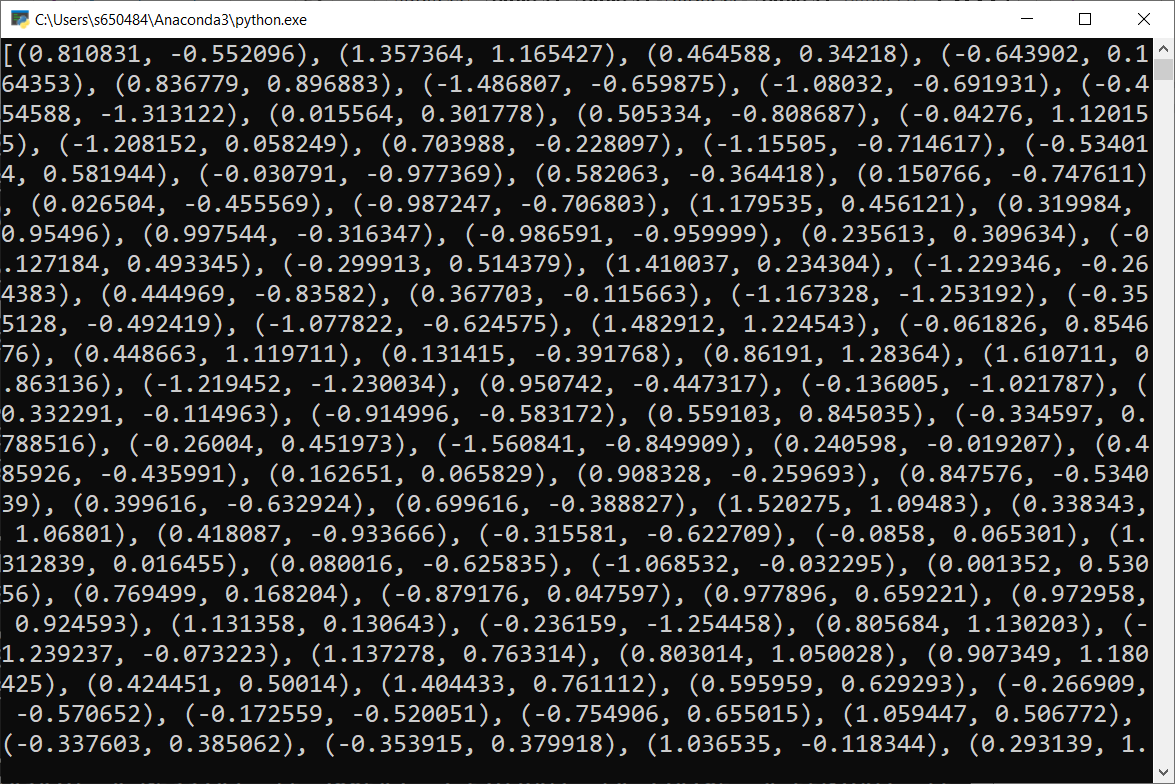
data\_list = iodata.read\_data\_file(**'ellipse1.txt**')

print(data\_list)

* Download **ellipse1.txt** and **ellipse2.txt** files from Canvas and place them in the same folder with .py files. Run the program. The data for **ellipse1.txt** is of the following form [(x1, y1), (x2, y2),. . , (xn, yn)]



* Change **ellipse1.txt** to **ellipse2.txt** and run your project again. The data has the same format



* The following code is to display the data on Canvas as a list of tuples (data samples) [(x1, y1), (x2, y2),. . , (xn, yn)]. Add the green highlighted code to the program in **Week4Tutorial.py** as seen below

import io\_data\_module as iodata

import tkinter

#Open file and read data

data\_list = iodata.read\_data\_file('ellipse1.txt')

print(data\_list)

#Create canvas

top = tkinter.Tk()

C = tkinter.Canvas(top, bg="white", height=700, width=700)

#Display data

for x, y in data\_list:

C.create\_oval(x-2, y-2, x+2, y+2, outline = "red", fill="red")

C.pack()

top.mainloop()

* Run your project and you will see all the data samples displayed in the top left corner on Canvas as below. The problem is the values of x & y in the data list are very small.



* To see the data samples clearly, we need to scale these x & y values. Add the yellow highlighted code to the program as follows

import io\_data\_module as iodata

import tkinter as tk

#Open file and read data

data\_list = iodata.read\_data\_file('ellipse1.txt')

#print(data\_list)

#Create canvas

top = tk.Tk()

C = tk.Canvas(top, bg="white", height=700, width=1000)

#Display data

s = 90 #scale factor

r = 4 #radius

for x, y in data\_list:

x = x\*s + 150 #some values are negative so +150 is to make them positive

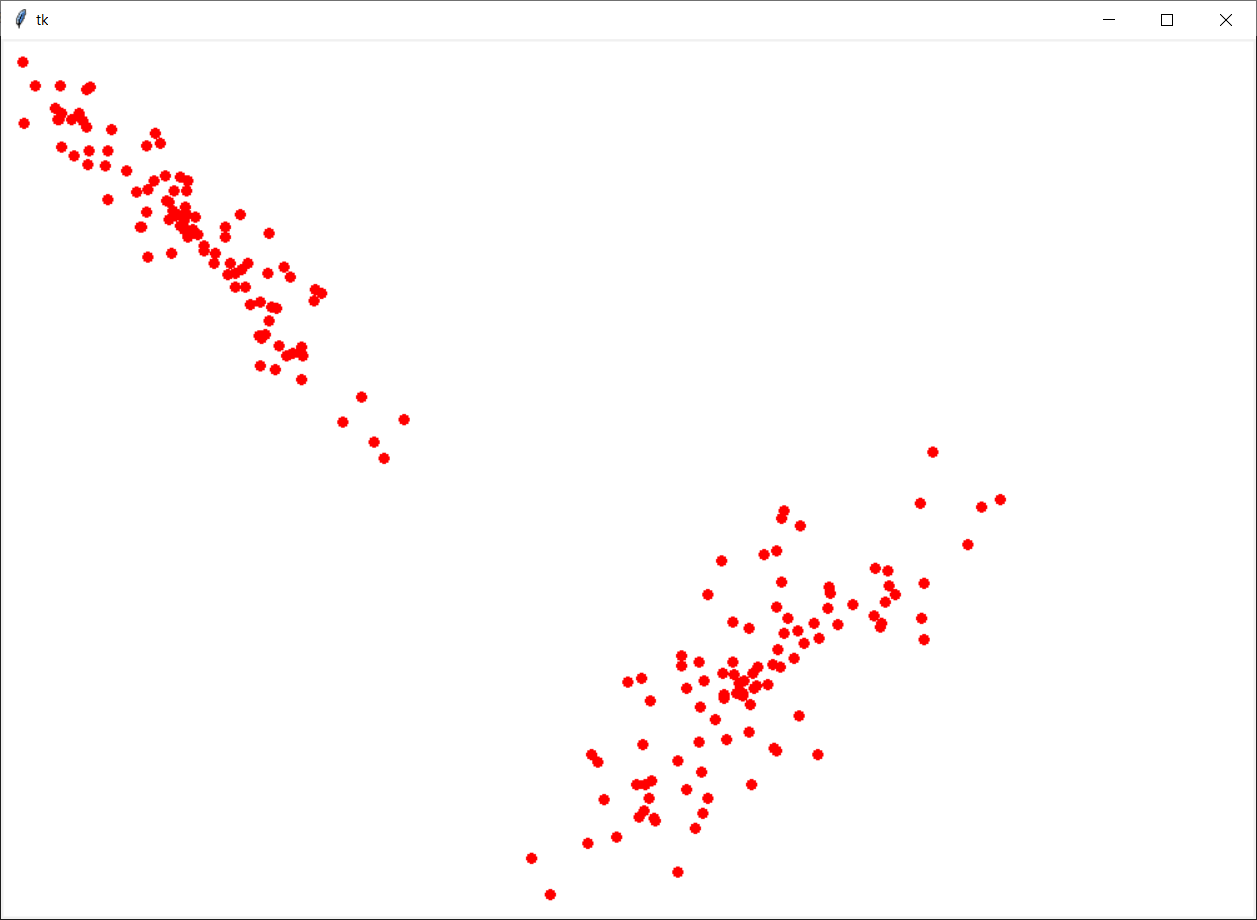
y = y\*s + 150

C.create\_oval(x-r, y-r, x+r, y+r, outline = "red", fill="red")

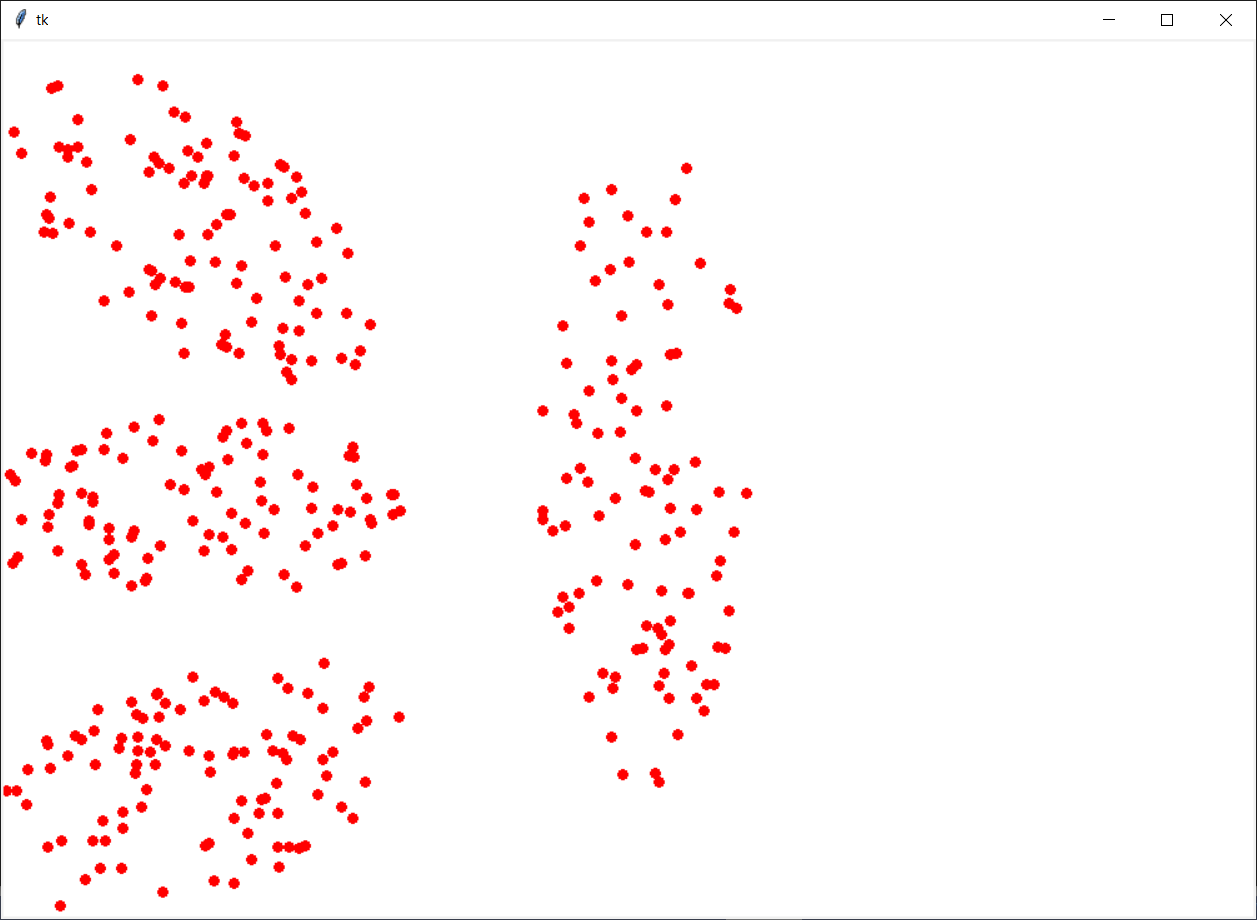
C.pack()

top.mainloop()

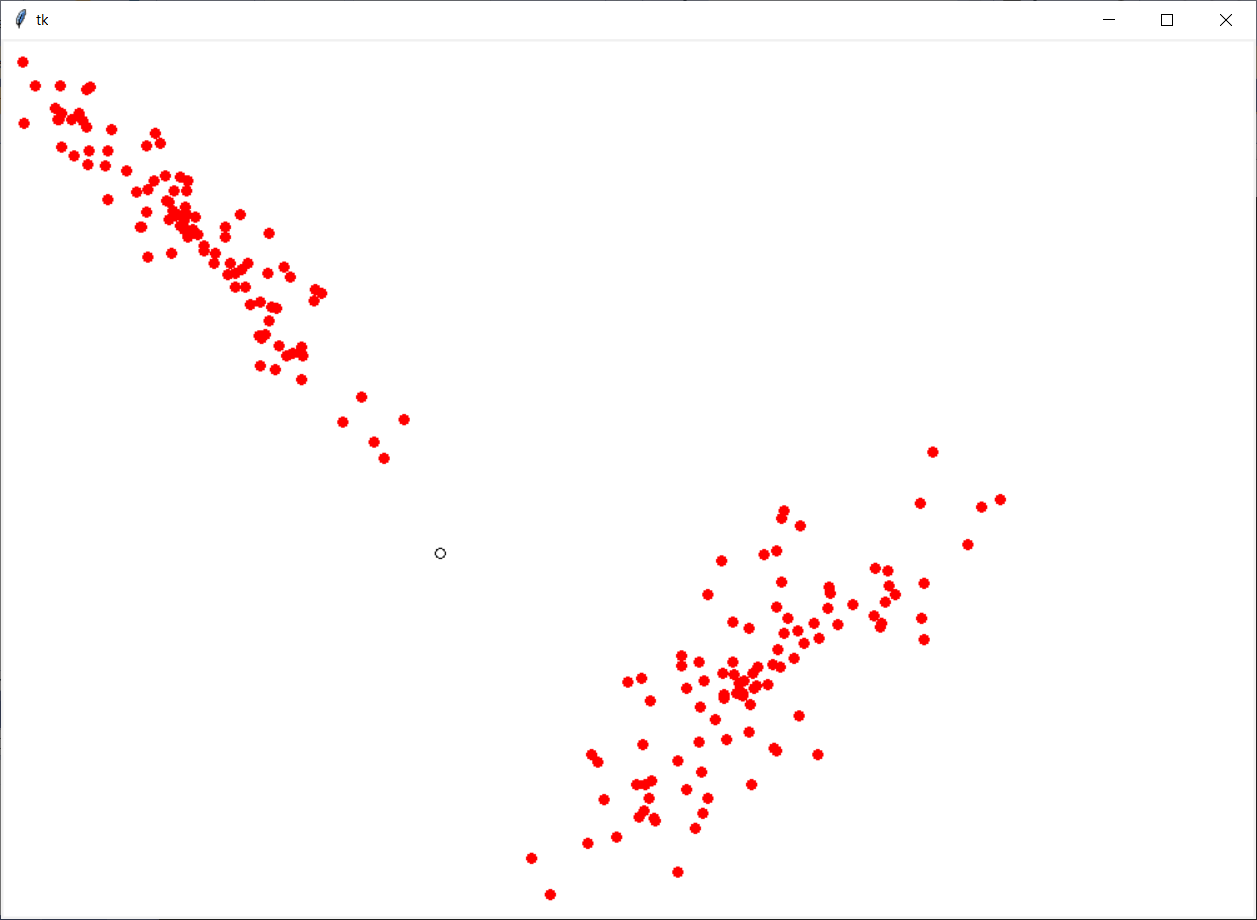
* Run the program again and we can see data samples as follows.



* Change the filename to ellipse2.txt and run the program.



**Question 1:** Modify your program (**Week4Tutorial.py**) to display the following data sample **unknown\_sample = (2.236779, 2.896883)** with the **ellipse1** dataset as follows



**Question 2:** Write a function named **find\_nearest\_neighbour** that takes **unknown\_sample** and **data\_list** as its input parameters and returns the *nearest* data sample of **unknown\_sample.** Place this function in **io\_data\_module.**

#define function

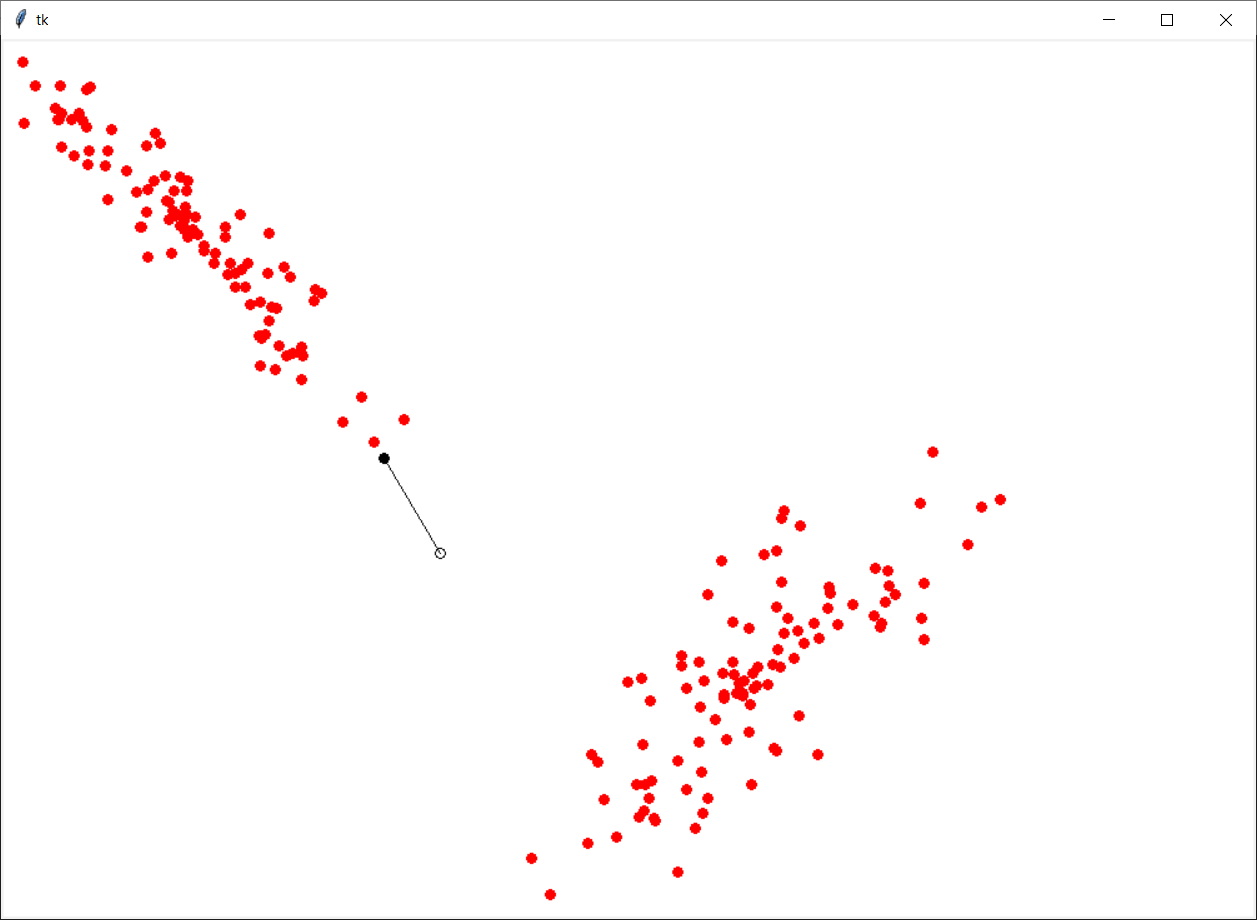
def find\_nearest\_neighbour(unknown\_sample, data\_list):

#write your code here

return nearest\_sample

#end function

**Question 3:** Modify your program (**Week4Tutorial.py**) to call the **find\_nearest\_neighbour** function to get the nearest sample of the **unknown\_sample**. Change colour of this nearest sample to black and draw a black line between this nearest sample and the unknown sample as follows



**Total mark for assessment: 3%. Complete and submit all examples and questions: 2%, and lab attendance: 1%. Submit after the due date: -0.5% and -0.5% for each day after. The due date is end of this tutorial session.**