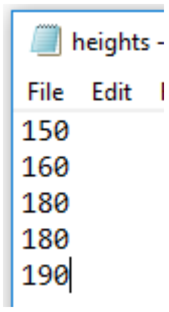
BREAKOUT ACTIVITIES

BREAKOUT 4.1: Data Processing (heights)

At the end of Section 2 we completed a breakout activity based on processing a number of height values input by the user. We start this activity by presenting a solution to the activity.

```python
# A program to calculate the average height of 5 people
# The heights are stored in a file called 'heights.txt'
heightFile = open("heights.txt","r") # Open the file

totalHeight = 0 # Initialise a running total to zero
height = float(heightFile.readline())    # read the first value
totalHeight = totalHeight + height       # keep a running total

height = float(heightFile.readline())    # read the next value
totalHeight = totalHeight + height       # keep a running total

height = float(heightFile.readline())    # read the next value
totalHeight = totalHeight + height       # keep a running total

height = float(heightFile.readline())    # read the next value
totalHeight = totalHeight + height       # keep a running total

height = float(heightFile.readline())    # read the next value
totalHeight = totalHeight + height       # keep a running total

# Calculate the average
avgHeigth = totalHeight/5

# Display the result
print("The average height is "+str(round(avgHeigth,2))+"cm")
print("The average height is", round(avgHeigth*0.393701,2), "inches")

heightFile.close()
```



```
150
160
180
180
190
```

The contents of the heights.txt file are shown here to the right. These can be edited and a new set of values supplied to the program at runtime.

We now add the following program into the mix. This three-liner uses the `mean` command from the `statistics` library to calculate the arithmetic mean of the numbers contained in the list `heightList`. If we wanted to calculate the mean of a different set of heights the programmer would need to change the values in this list.

```python
from statistics import mean

heightList = [150, 160, 180, 180, 190]
print(mean(heightList))
```

It is worth browsing to https://docs.python.org/3/library/statistics.html to take a look at the official Python documentation on the statistics package.

Suggested Activities

1. Compare and contrast the two programs

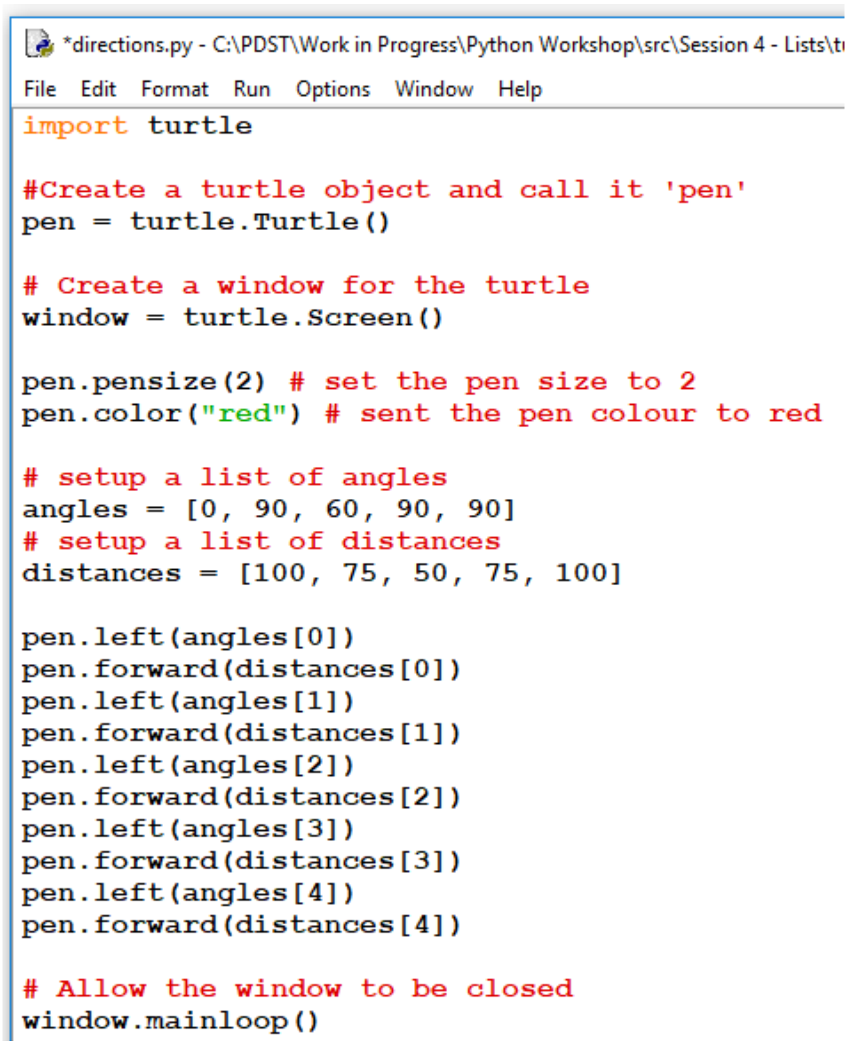
2. Modify the first program so that:
   - it uses the `statistics` package to calculate the mean. (The variable `totalHeight` should not appear in the final program.). Test your code to make sure it gives the same result as the original code.
   - instead of there being five separate uses of the `readline` command, the entire file should be read into a string by single `read` command.
   - It uses the `split` command to create a list of height values from the resulting string. (You may wish to look back at the fruit machine v2 example for some technical guidance to complete this activity).
   - The program should be able to deal with a variable number of heights and as well as decimal values.

3. Display the following information in meaningful messages (preferably using string formatting to display any variable data):
   - The median
   - The maximum height
   - The minimum height
   - The range

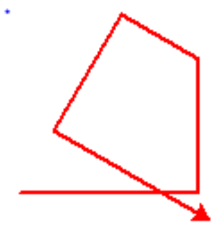
BREAKOUT 4.3: Turtle Directions

The purpose of this activity is to build upon the knowledge we already have about turtle graphics and apply the concepts from this section to them.

The turtle graphic program shown below contains two lists – angles and distances. Between them the lists contain 'encoded' data that direct a turtle from point A to point B. The program listing is as follows:


*directions.py - C:\PDST\Work in Progress\Python Workshop\src\Session 4 - Lists\t
File Edit Format Run Options Window Help

```
import turtle

#Create a turtle object and call it 'pen'
pen = turtle.Turtle()

# Create a window for the turtle
window = turtle.Screen()

pen.pensize(2) # set the pen size to 2
pen.color("red") # sent the pen colour to red

# setup a list of angles
angles = [0, 90, 60, 90, 90]
# setup a list of distances
distances = [100, 75, 50, 75, 100]

pen.left(angles[0])
pen.forward(distances[0])
pen.left(angles[1])
pen.forward(distances[1])
pen.left(angles[2])
pen.forward(distances[2])
pen.left(angles[3])
pen.forward(distances[3])
pen.left(angles[4])
pen.forward(distances[4])

# Allow the window to be closed
window.mainloop()
```



When the program is run it displays the path show.
Experiment by changing the values of the angles and distances to see what journeys you can come up with.
(Remember that the turtle starts in the centre of the screen facing right (east). The arrowhead indicates the end of the path.)

Task 1.

Modify the code so that it:

- ***draws a 100 x 100 blue square***
- ***draws a green hexagon (side length = 75)***
- ***draws a yellow equilateral triangle of length 50 (hint: you may need to change the angles list)***

Task 2.

Modify the code so that it:

- ***randomly selects an angle from the angles list for the first angle***
- ***randomly selects a distance from the distances list for the first distance***
- ***randomly selects an angle from the angles list and a distance from the distances list each time***