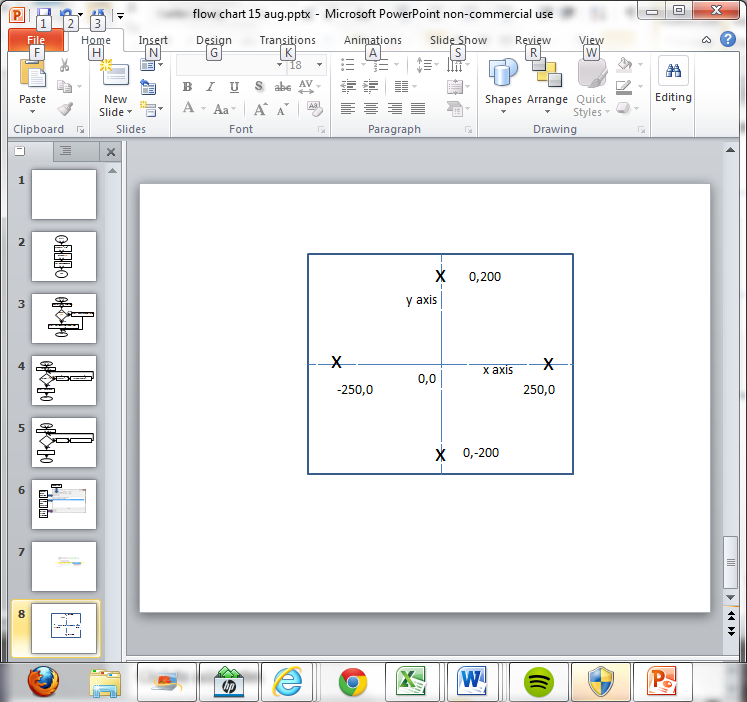
**Activity 22.1**

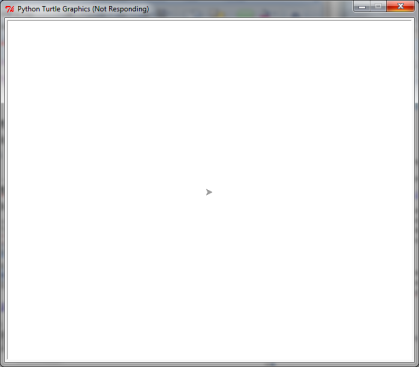
**Turtle graphics in Python (programming using x,y Cartesian coordinates)**

Cartesian co-ordinates is a way of specifying an exact point using two values (x,y).

Turtle graphics allows a robotic turtle to be programmed to move around in an x,y plane “playground” graphic canvas. The turtle starts at x,y co-ordinates (0,0) in the centre of the graphics canvas. The turtle holds a pen which can be placed up or down. When down the pen ‘draws’ on the canvas. By changing the direction and moving the turtle around, you can draw shapes and pictures.

Turtle graphics is an example of object oriented programming. Each turtle is an object which has properties such as turtle position (x,y) turtle heading (a bearing in degrees), turtle shape (turtle, arrow, circle, etc.), pen (up or down), pen colour (‘colour’), pen width. There can be more than one turtle on the canvas.

**Exploring turtle graphics**

* Create a turtle object using the commands:

>>>import turtle

>>>t = turtle.Pen()

Where t is the name of the variable that refers to the turtle.

This should display the graphic canvas with the turtle at position (0,0). (The turtle is the arrow shape in the middle of the screen.)

**To alter the turtle use:**

**Turtle\_name=turtle.command() e.g. t=turtle.forward(50)**

**To view a property of the turtle use**

**Turtle\_name.function() e.g. t.position()**

* Explore these commands:
  + To change the shape to a turtle

>>>t= turtle.shape("turtle")

* + To change the colour of the turtle

>>>t=turtle.color("blue")

* + To move turtle

t=turtle.forward(50)

* + To turn turtle

>>>t=turtle.right(90)

* + To put the pen down

>>>t=turtle.pendown()

* + To change the pen colour

>>>t=turtle.pencolor("red")

* + To view the position of the turtle

>>>t.position()

* + To view the heading of the turtle

>>>t.heading()

**Programming challenges using turtle graphics**

* Write a program that draws a square.

t = turtle.shape("square")

* Write a program that draws four squares in different colours at different locations on the canvas
* Write a program that draws a circle.

t = turtle.shape("circle")

* Write a program that draws a square with a circle inside.
* Write a program that draws a square where each side is a different colour.
* Write a program that ‘stamps’ a different colour of the turtle in each of the four corners of the canvas.
* Copy and run this function. Explain the purpose of the first three lines. What does the function do?

def t2():

import turtle

t=turtle.Pen()

t.reset()

t.goto(10,10)

t.pencolor("blue")

t.width(5)

t.circle(70)

t.goto(10,10)

t.pencolor("pink")

t.circle(50)

* Draw two more circles in different colours all starting at 10,10.
* Copy and run this function. What does it do? Try altering some of the values to see what kind of pattern it will draw.

def t1():

import turtle

t=turtle.Pen()

t.reset()

for x in range(1,20):

t.forward(100)

t.left(95)

For more details on turtle graphics see <http://docs.python.org/2/library/turtle.html>

**Activity 24.1**

**Records and files**

A file is made up of records (or lines) in order. Files are sequential which means you have to read each record (or line) in order to get to the next record. A file pointer is used to keep track of which record is currently being processed.

|  |
| --- |
| Record 0  File pointer |
| Record 1 |
| Record 2 |
| Record 3 |
| Record 4 |

A file has a file name (for example myFile.txt) and a file access mode. The file access mode indicates how you want to access the file.

|  |  |
| --- | --- |
| **File access modes** | **Explanation** |
| r | Opens a file for reading. File pointer is set to 0. |
| w | Opens a file for writing. The file is overwritten if it exists so TAKE CARE!  If the file does not exist a new file is created. |
| a | Opens a file for writing by appending to the end. File pointer is set to the end of the file. If the file does not exist a new file is created. |

When using files in Python you have to open the file and close the file. The close command makes sure all information is written to the file. If strange things are happening check you have closed the file.

**Directory structures and files**

The default file location is the Python installation directory (often c:Python32). This is where Python will expect to find files and will store files. If you want to use other directories use the path name for that directory.

1. **Writing a record to a file**
   * Copy and run the program below which writes records to a file which is called “NewFile.txt”.

defwriteToFile():

myFile = open("NewFile.txt","w")

for each in range(1,8):

record = "This is record number {0} in the file \n".format(each)

print(record)

myFile.write(record)

myFile.close()

* + Open the file (in Notepad) and make sure it has written to the file. The newline escape code “\n” is put at the end of the line so that each line is stored as a separate record.
  + Amend the program to write your name at the end of each record.

*Hint: You can only write strings to a file.*

1. **Reading a record from a file**

* Copy and run the program to read the records from the file you created above.

defreadFromFile():

myFile = open("NewFile.txt","r")

for each in range(1,8):

record = myFile.readline()

print("this is record {0} {1}".format(each, record))

myFile.close()

* Amend the program to write four lines of your favourite song lyrics to a file and read them back.

**Activity 27.1**

Use Python to experiment with how the computer handles strings, floats and bools. What is the outcome of running each of these code snippets? Why does this happen?

**#Snippet 1**

Snippet1

|  |  |
| --- | --- |
| Outcome: | File "//DC01-FILE-001/kaloi002.309$/Documents/Computer Science/Python/Word Files/computerhandlesstring.py", line 3, in <module>  print(Number1 \* Number2)  TypeError: can't multiply sequence by non-int of type 'str' |
| Why did this happen? | It cannot multiply non-intergers/strings. |

**#Snippet 2**

Snippet2

|  |  |
| --- | --- |
| Outcome: | Enter your first number:isaac  Enter your second number:6  isaacisaacisaacisaacisaacisaac |
| Why did this happen? | Because if you multiply a string with an integer, it’ll only display it the amount of times the second number is. |

**#Snippet 3**

Snippet3

|  |  |
| --- | --- |
| Outcome: | Enter your first number:1  Enter your second number:2  2 |
| Why did this happen? | It times the integer with another integer. |

**#Snippet 4**

Snippet4

|  |  |
| --- | --- |
| Outcome: | Enter your first number:3  Enter your second number:2  6.0 |
| Why did this happen? | It displays the decimal – that what Float is/does. |

**#Snippet 5**

Snippet5

|  |  |
| --- | --- |
| Outcome: | Enter your first number:1  Enter your second number:3  3.0 |
| Why did this happen? | Because ‘float’ is involved it’ll add a decimal point to the 3 meaning, the answer will have to have a dp. |

**#Snippet 6**

Snippet6

|  |  |
| --- | --- |
| Outcome: | Traceback (most recent call last):  File "//DC01-FILE-001/kaloi002.309$/Documents/Computer Science/Python/Word Files/computerhandlesstring.py", line 3, in <module>  print(Number1 \* Number2)  TypeError: can't multiply sequence by non-int of type 'float' |
| Why did this happen? | You can’t multiply a number (with a dp) to a string. |