1. Given a line of text (i.e., a string terminated by a ‘\n’ character), write a function that returns the list of words contained within that line. For example,

```python
line = '   2the, Line-of-text444a    ONe of ++four fox\n'
print findWords(line)
```

should print: ['the', 'Line', 'of', 'text', 'a', 'ONe', 'of', 'four', 'fox']

a) What is a good definition of a word?

b) If we knew the position in line where the word started and the position of first character after the word, how could we extract the string for the word?

c) Let's develop an algorithm for the function findWords.
2. Turtle Graphics is discussed in sections 7.1 and 7.2 of the textbook and is NOT a standard module of Python. The objectives for looking at the Turtle graphics module are:

- Gain experience using object-based programming using existing classes, objects, and methods (p. 250)
- Understand simple graphics operations to draw 2D shapes

The following program (Figure 7.4) causes the turtle to move 30 times for a distance of 20 pixels each in a random direction with each move drawn in a random color.

```python
from turtlegraphics import Turtle
import random

def randomWalk(turtle, turns, distance = 20):
    turtle.setWidth(1)
    for x in xrange(turns):
        turtle.turn(random.randint(0, 360))
        turtle.setColor(random.randint(0,255),  # red component
                         random.randint(0,255),  # blue component
                         random.randint(0,255))  # green component
        turtle.move(distance)

randomWalk(Turtle(), 30)
temp = raw_input("Hit the <Enter> key to quit")
```

Define the following functions:

- drawLine - this function expects a Turtle object and four integers as arguments. The integers represent the end-points of a line segment. The function should draw a black line segment of width 1 with the turtle and do no other drawing.

Write a main function that uses drawLine such that:

- creates a Turtle with a window size of 400 by 400.
- draws a line from (150, 150) to (-200, -200).