Hangman in Python

In this project we'll be creating a simple hangman game in Python, using ASCII characters to print out the pictures

> If you haven't got Python installed on your machine, follow instructions at https://www.python.org/downloads/

Setup

First we'll import the `random` package so we'll be able to randomise the word chosen

```
import random
```

Then we'll set up a list of words, a variable that selects a random word, and a set of characters the user can input (only letters)

```
wordList = ("phone", "laptop", "desktop", "television", "never", "g
uess", "nice", "chair", "car")
word = random.choice(wordList)
acceptable = ("abcdefghijklmnopqrstuvwxyz")
```

We'll also set up variables to store the stats during the game

```
guessed = []
state = 0
hasWon = 0
playedOnce = 0
```

The Main Function

Here we'll define the main function of the program (including calls to the rest of the functions we'll write)

```
def main():
    global guessed, hasWon, state, playedOnce, word, wordList
    setup_game()
    newPrint("My word is " + str(len(word)) + " letters long.")
    while (wantsToPlay() == 1):
        word = random.choice(wordList)
        guessed = []
        playedOnce = 1
        hasWon = 0
        state = 0
        while (hasGuessed() == 0 and state < 7):
            drawStickman()
            drawWord()
            takeNewLetter()
        drawStickman()
        newPrint("My word was " + word)
```

Breaking down the above :

- First we get the variables from the `global` scope
- We run `setup_game()` and then output the length of the word to guess
- Inside the game loop
    - A random word is chosen
    - The variables are all reset
    - Then while the player hasn't guess the whole word, we continue to take new letters and draw the appropriate stickman
    - At the end of the game loop the word is revealed

Defining other Game Functions

wantsToPlay()

We have a function to allow the user to re=play the game once it has ended, asking for yes or no input

```
def wantsToPlay():
    if (not playedOnce):
        return 1
    l = input("\nWould you like to play again? (y/n)")
    while (l != "y" and l != "Y" and l != "n" and l != "N"):
        l = input("\nWould you like to play again? (y/n)")
    if (l.lower() == "y"):
        return 1
    return 0
```

First we check if the user has already played - if not the game automatically starts

We can see that the loop continues to ask the question until `l` is either `y` or `n`. Note also that `l.lower()` is used to process either `y` or `Y` in the same way

takeNewLetter()

We have a function that takes the next letter from the player

```
def takeNewLetter():
    global state, hasWon
    newPrint("So far, you have guessed the following letters...")
    for g in guessed:
        print(g, end=" ")
    letter = input("\n\nWhat letter would you like to guess next?
\n")
    while (letter in guessed or letter not in acceptable):
        if (len(letter) > 1):
            if (letter.lower() == word.lower()):
                 newPrint("You win!")
                 hasWon = 1
                 break
            else:
                newPrint("Boo... that was wrong... you're dead...")
                state = 7
                break
        else:
            if (letter not in acceptable):
                letter = input("That character is unacceptable. You
many only enter lower case letters.\n")
            else:
                letter = input("You have already guessed that lette
r, try another one...\n")
    guessed.append(letter)
    if (letter not in word):
        state += 1
    return
```

- First the function prints out all the letters that have been guessed already
- It then gets a new guess from the player
- Below the `while` loop we can see that the letter is added to the list of guesses, and the state is increased if the letter was not in the word
- Inside the `while` loop we see two cases
    - The player guesses a whole word, in which they either win or lose the game completely
    - The player guesses a single character, in which case we check if it's an acceptable letter and if it's already been guessed

drawWord()

`drawWord()` lets the player see the current word, with already guessed letters shown

```
def drawWord():
    tempWord = ""
    for c in word:
        if (c in guessed):
            tempWord += c + " "
        else:
            tempWord += "_ "
    newPrint(tempWord)
    return
```

The entire word is looped through, with either the letter or a `-` being printed depending on if the letter was guessed already

drawStickman

This function draws the appropriate stickman depending on the number of incorrect letters guessed (indicated by the `state` variable)

```
def drawStickman():
    if (state >= 7):
        print("   _______")
        print("|/      |")
        print("|      (_)")
        print("|      \|/")
        print("|       |")
        print("|      / \\")
        print("|")
        print("|___")
        print("Oops. You're dead.")
    elif (state == 6):
        print("   _______")
        print("|/      |")
        print("|      (_)")
        print("|      \|/")
        print("|       |")
        print("|      / ")
        print("|")
        print("|___")
    elif (state == 5):
        print("   _______")
        print("|/      |")
        print("|      (_)")
        print("|      \|/")
        print("|       |")
        print("|")
        print("|")
        print("|___")
    elif (state == 4):
        print("   _______")
        print("|/      |")
        print("|      (_)")
        print("|      \|/")
        print("|")
        print("|")
        print("|")
        print("|___")
    elif (state == 3):
        print("   _______")
        print("|/      |")
        print("|      (_)")
        print("|      \|")
        print("|")
        print("|")
        print("|")
        print("|___")
    elif (state == 2):
```

```
            print("   _______")
            print("|/      |")
            print("|      (_)")
            print("|")
            print("|")
            print("|")
            print("|")
            print("|___")
        elif (state == 2):
            print("   _______")
            print("|/      |")
            print("|")
            print("|")
            print("|")
            print("|")
            print("|")
            print("|___")
        elif (state == 1):
            newPrint("As this is your first mistake, I will let you of
f...")
            print("   _______")
            print("|/")
            print("|")
            print("|")
            print("|")
            print("|")
            print("|")
            print("|___")
        elif (state == 0):
            print("   _______")
            print("|/")
            print("|")
            print("|")
            print("|")
            print("|")
            print("|")
            print("|___")
```

hasGuessed()

The function is used by the `main` function to check when the game loop needs to be broken. It returns a `1` when the game should finish

```
def hasGuessed():
    if (hasWon == 1):
        return 1
    if (state >= 7):
        return 1
    for c in word:
        if (c not in guessed):
            return 0
    if (len(guessed) == 0):
        return 0
    return 1
```

setup_game()

The function simply prints out some lines at the start of the game

```
def setup_game():
    newPrint("Welcome to the Hangman game!")
    newPrint("I have chosen a random word from my super secret lis
t, try to guess it before your stickman dies!")
```

newPrint()

The `newPrint()` function is used to print out each message on a new line

```
def newPrint(message, both = 1):
    msg = "\n" + message
    if (both != 1):
        msg += "\n"
    print(msg)
```

Beginning the Game

Finally, we'll start the main function and output a message at the end

```
main()
newPrint("Thank you for playing.")
```

Running the Game

To run the game in your console simply run it in the command line -

```
python hangman.py
```