Name: Alex Barker
ID: U1512136

Assignment 2 Descriptive/Notes

A1.1.1

Due to there only being 2 cases (starting with vowel or not starting with vowel) and the case where a word starts with a vowel can be written in 1 line (ouput = input + "hay"). This saved effort and time, allowing the code to assume any input doesn't start with a vowel, iterate for that then check after if it did and redo the output if it did in fact start with a vowel.

A1.1.1c1

Not being able to do the old trick of redoing the output in one line if it did in fact start with a vowel due to the input having to have punctuation removed. This meant that more lines had to be added to make sure for the first letter being a vowel the process would be carried out correctly, removing the (output = input + "hay"). This was checked by "if word[0] in vowels and isFirst:".

A1.1.2

I thought it would be appropriate these as functions as they seem to be something that you would want to call with changing variables. These are then printed at the bottom outside the functions with the values asked for: Tangent(0,89), VowelsExtract("Alex Barker Assignment 2"), equation(-5,5,10). It would have been possible to create a 'YMin' variable for this although it would add unnecessary lines as the function never dips below 5 which is above the 0 minimum specified.

A1.1.3.1

IN: [x**2/16 - y**2/4 for x in range (0,5) for y in [4,8,12,16,20]]
OUT: [-4.0, -16.0, -36.0, -64.0, -100.0, -3.9375, -15.9375, -35.9375, -63.9375, -99.9375, -3.75, -15.75, -35.75, -63.75, -99.75, -3.4375, -15.4375, -35.4375, -63.4375, -99.4375, -3.0, -15.0, -35.0, -63.0, -99.0]

This works by using x-values from 0 to 5 in steps of 1 by default for the y-values [4,8,12,16,20]. The function's BIDMAS order is quite important, it squares both x and y-values then divides them by 16 and 4 respectively, it then subtracts the y**2/4 value from the x**2/16 value. It selects the order of values as follows. It will run through all the x=0 values against the y=4,8,12... then increase x by the step (not defined so default 1) so therefore, move onto x=1 for y-values of 4,8,12...

for loop equivalent:

```
List = []
for x in range (0,5):
   for y in [4,8,12,16,20]:
      List.append(x**2/16-y**2/4)
print(List)
```

A1.1.3.2

.py file fairly simple

A1.1.3.3

The list comprehension given gets the numerical value from each entry which has the first part equal to 'John' for each key. It then sums these and returns the total for every mark that the entry 'John' has.

A1.1.3.4

The duplicates list comprehension uses the pop function to remove values from list 1 and add them to the entry of the created duplicates list. It iterates for each value in list1 checking if that value exists anywhere in list 2. The problem is if it removes a value in the list then increasing the iteration(i) value by 1. This leads to on the next iteration skipping the next value since they have all been shifted to the left and the i value is now not representative of the order which it should be checking.

This could be changed to:

```
duplicates = [list1.pop(list1.index(i)) for i in list1[::-1] if i in list2]
```

The "for i in list1[::-1]" reverses the order that the comprehension iterates over, therefore if a value is removed the i values still stand for decreasing tests.

A2.1

Input loop moved to outside of function, wasn't sure about punctuation as new rule says all non characters are ignored, therefore my new code treats punctuation the same way as numbers now and 'puncFound' list has been removed as punctuation will fall into the "else: #symbols and numbers" category.

Name: Alex Barker
ID: U1512136

A2.2

A basic Fibonacci sequence, example given uses an input of 100 into the function, thus printing to the next largest (144), a few lines were added for the specific case of a 0 or a 1 input. I believe [0] and [0,1] are the appropriate output for this.

A2.3

Offsets were the biggest challenge here, once that was worked out everything fell into place.

A3.4

To inherit from a base class “super(self, word).__init__()” could be used to invoke something such as the constructor of the base class from within the “gibberish_converter” method in the inherited class.