**Insertion Sort vs Bubble Sort by Sam Twite**

**Bubble Sort**

import random

import time

start\_time = time.time()

value = 0

numbers = []

for i in range(1000):

    value = random.randint(0,100000)

    numbers.append(value)

def bubble\_sort(numbers):

    swapped = True

    while(swapped):

        swapped = False

        for i in range(len(numbers) - 1):

            if numbers[i] > numbers[i+1]:

                #swap

                numbers[i], numbers[i+1] = numbers[i+1], numbers[i]

                swapped = True

bubble\_sort(numbers)

print(numbers)

print("time taken:")

print((time.time() - start\_time))

**Insertion Sort**

import random

import time

start\_time = time.time()

value = 0

numbers = []

for i in range(1000):

    value = random.randint(0,100000)

    numbers.append(value)

def insertionSort(numbers):

    for i in range(1, len(numbers)):

        key = numbers[i]

        j = i-1

        while j >=0 and key < numbers[j] :

                numbers[j+1] = numbers[j]

                j -= 1

        numbers[j+1] = key

insertionSort(numbers)

print(numbers)

print((time.time() - start\_time))

**Time comparisons**

For 1,000 numbers:

Bubble: 0.19 seconds

Insertion: 0.0479 seconds

Insertion sorting was far quicker however to see a greater disparity I made the data set larger and timed it again for 10,000 integers numbers.

Bubble: 11.667 seconds

Insertion: 4.099 seconds

In this test it was obvious that insertion sort was much quicker beating the bubble sort time by over 6 seconds, if I were to increase the size of the data set even more, the difference in time would only increase.