2. What is the running time, T(n), for the below code? Describe how you arrived at the answer (which part of the code correspond to the terms in T(n) and why). Determine the asymptotic upper bound. What is **C, n_0** in your upper bound? Plot **n vs T(n)** and **Upper bound** in the same graph for n = 1000 to 5000, in increments of 1000. **[15 points]**

$O(n^2)$

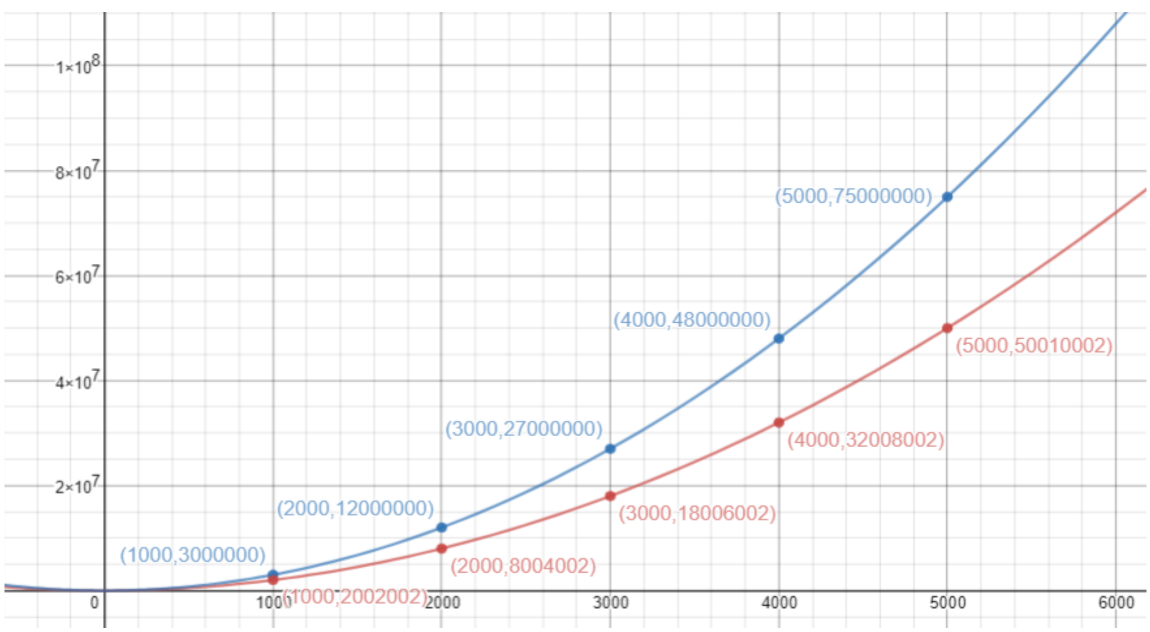
```
def random_func1(list1, list2):

  x = 0
  g = 0
  while(x < len(list1) + len(list2)):
   for i in range(len(list1)):
      print(list1[len(list1)-1-i])
      g = g + 1
    y = 0
    while(y < len(list2)):
      print(list2[y])
      y +=2
    x += 1
```

$\underline{2n^2 + 2n + 2}$

$(2n + 2)n$

The running time, T(n), is $2n^2 + 2n + 2$. I got this answer by first calculating the value inside the while loop outside, which equals 2n+2. This was calculated by giving the assignments in the nested loops n value and 1 to the ones outside of the loops. Then I multiplied (2n+2)*n to get $2n^2 + 2n + 2$. So if T(n) can be described as $2n^2$ then the asymptotic upper bound could be $3n^2$. This would make the C = 3 and the $n_0 \approx 2.732$. This function is parabolic and can be written in big-O notation as $O(n^2)$. In the graph below, the upper bound is represented by the blue line and plots, the T(n) function is represented by the red line and plots. The n_0 is not visible because of the scale of the graph. X-axis=n, Y-axis=T(n)



3. What is the running time, T(n), for the below code? Describe how you arrived at the answer (which part of the code correspond to the terms in T(n) and why). Determine the asymptotic upper bound. What is **C, n₀** in the upper bound? Determine the asymptotic upper bound. Plot **n vs T(n)** and **Upper bound** in the same graph for n = 10 to 1000, in increments of 10. **[15 points]**

$T(n)=n^3$

$O(n^3)$

```
for i in range(n):
  for j in range(n):
    for k in range(n):
      k = 2 + 2
```

1 n n^2 n^3

The running time, T(n), is n^3. I got this answer by calculating the value in the innermost loop, which is 1 executed n times. You now have n within the middle loop. So, you multiply it by n. Now, you have n^2 within the outside loop. So, you multiply it by n again giving you n^3. An asymptotic upper bound could be $2n^3$. The C of this function is 2 and the n_0 is 0. This function is cubic and can be written in big-O notation as $O(n^3)$. In the graph below, the upper bound is represented by the red line and plots, the T(n) function is represented by the blue line and plots. The n_0 is the black point at (0,0). X-axis=n, Y-axis=T(n). I labeled every 200n due to space concerns.