**Time and Space Complexity of the RSA Algorithm**

**Key Generation**

1. **Miller-Rabin:**

**A screen shot of a computer program

Description automatically generated**

* + **Time Complexity**: O(n^4)

This is because we call Mod\_exp which has a complexity of O(n^3) and we do this n times as we are removing one bit each time.

* + **Space Complexity:** O(n^2)

This is inherited from Mod\_exp as the memory is overwritten each iteration.

1. **Mod\_exp**:

A black screen with colorful text

Description automatically generated

* + **Time Complexity**: O(n^3)

This is where n is the largest number of bits of the 3 input numbers. At most, we will have n recursive calls and each call will multiply 2 n-bit numbers.

* + **Space Complexity**: O(n^2)

The inputs are each O(n) and it will have at most n recursive calls

1. **Fermat function:**

A screen shot of a computer program

Description automatically generated

* + **Time Complexity**: O(n^3).

This is because mod \_exp has a complexity of O(n^3) and, the loop will be disregarded as it is k times, which in the rules for big O, O(k\*n) is the same as O(n) if k is a constant.

* + **Space Complexity:** O(n^2)

The space complexity is again O(n^2) because there aren't any significant space requirements for the algorithm itself, except for that which is inherited from Mod\_exp function

1. **Ext\_euclid function:**

**A computer screen with text and numbers

Description automatically generated**

* + **Time Complexity**: O(n^3)

This is because the modular division is O(n^3) as each return is O(n^2) and we do this n times

* + **Space Complexity:** O(n^2)

For each N stack of the recursion, you store a number of n size.

1. **Gen\_large\_primes function:**

**A computer screen shot of a program code

Description automatically generated**

* + **Time Complexity**: O(n^4) while using Fermat and O(n^5) for Millar-Rabin

This is because using each of the functions have their respective time complexity and we do this at most n times

* + **Space Complexity:** O(n^2)

This is inherited from Mod\_exp which is in both respective functions

1. **Gen key pair’s function:**

**A computer screen with text and numbers

Description automatically generated**

* + **Time Complexity:** O(n^4)

This is because we generate 2 large primes inheriting the time complexity of that function and we call at most Ext\_euclid e times which gives us 2n^4 + en^3 which gives us n^4

* + **Space Complexity:** O(n^2)

This is inherited from Fermat which is used in the generating of the large prime numbers

**Probability**:

A screen shot of a computer code

Description automatically generated

Fermat: The likelihood of a false positive decreases by half with each test iteration. Subtracting this value from 1 gives the probability of the result of being correct.

Miller-Rabin: Similarly, the probability of a false positive is ¼ and using the same logic we can use 1- ¼^k probability of being correct