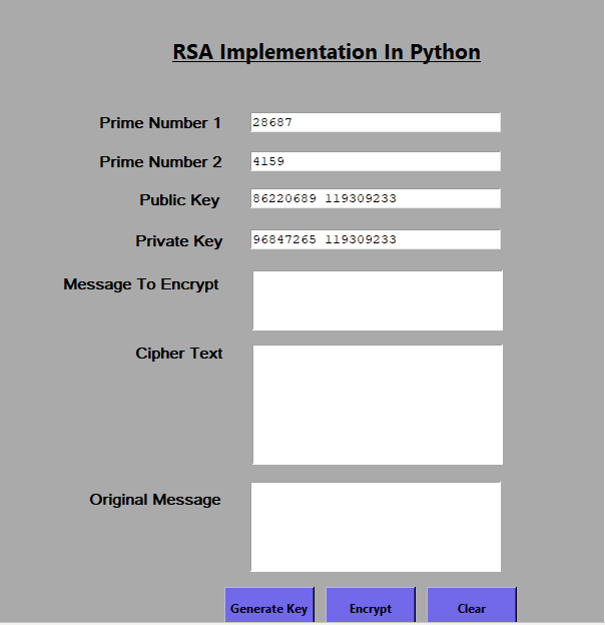
# Overview of the Project

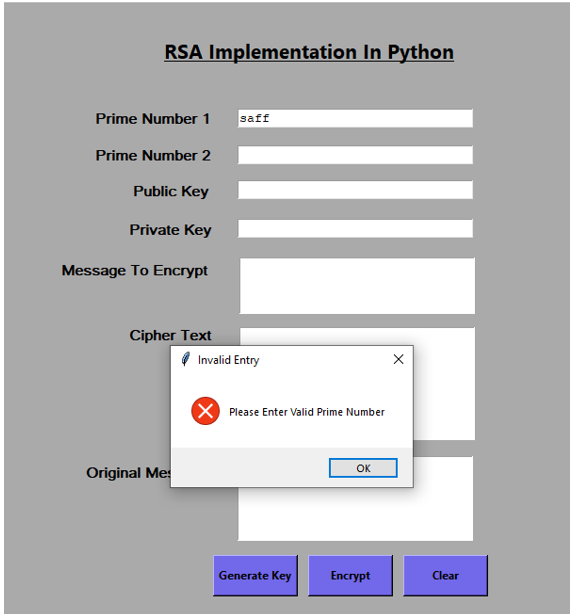
This project is a GUI based desktop application, which is capable of accepting two prime numbers greater than 10 and generate public and private key of RSA encryption algorithm. My application also capable of getting message as user input and Generating cypher text using public key and again decrypt the cipher text back to original message by using private key. This GUI application is capable of validating user inputs and generate message when user insert invalid inputs into entry fields. I am using python tkinter library for creating user interface in this project.

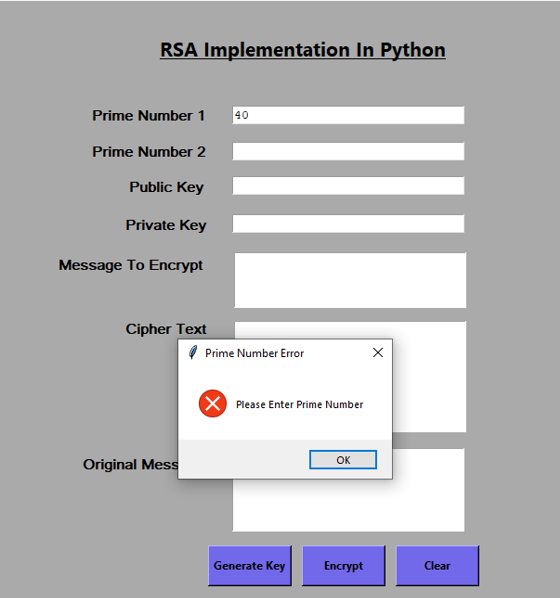
# Input and Output screenshots

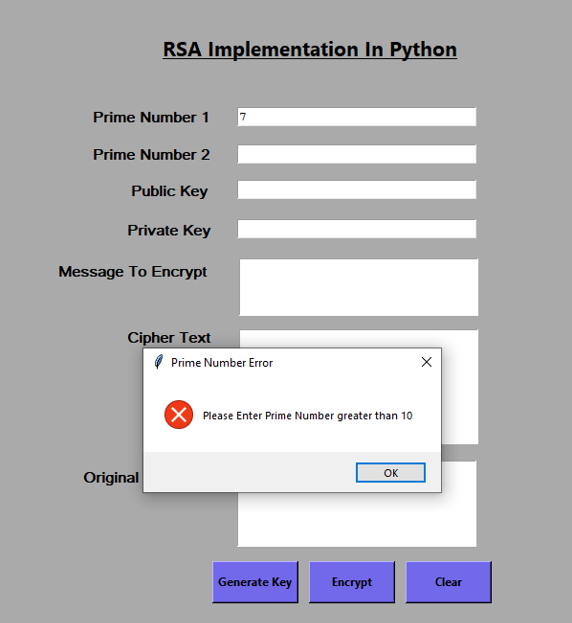
## Key generation

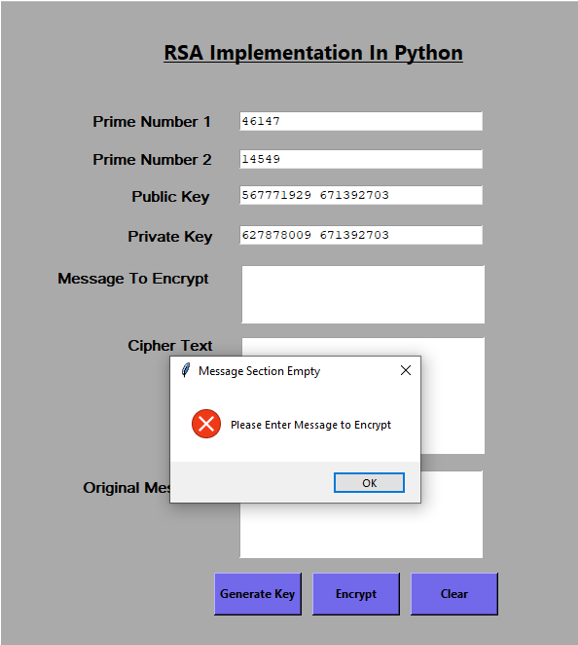


## User input validation

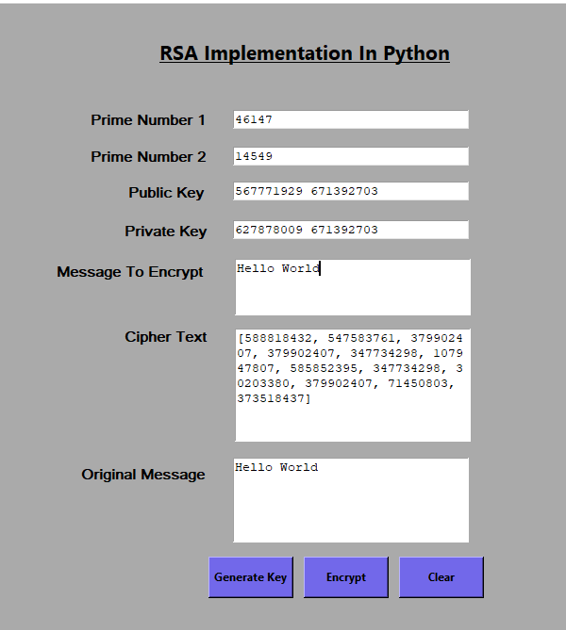








## Message encryption and decryption



# Project File structure

I have 4 python files in the project. Main.py file contains starting code of the GUI application. I have separated logic and view into separate files for easy maintenance and code reusability.Crypto.py file contains Rsa class which handles all RSA cryptography related functions. Model.py file contains Logic class which handles all button click events in the project. View.py file contains Interface class which handles user interface creation.

# Crypto.py file

Outside the Rsa class I am calculating and assigning a maximum number system can handle by using sys.math function and that max value will be used in the Rsa class.

## Rsa class

### Imod method

This function is helps to compute d=e-1 (mod n) function and calculate d value. This function gets e and n as inputs. D and e are called modular inverse. I am using Euler's theorem to calculate modular inverse.

### Is\_prime method

This functions is responsible of checking given number is a prime number or not. This function accept number and return Boolean value.

Following steps explain how algorithm works in is\_prime methos.

1. Check given number is least prime number 2 or not. If it is 2 then return true.
2. Check if given number is divisible by two without having any remainder or if given number is less than or equal to 1 then return false.
3. Get square root of the given number and eliminate decimal points and add one to that. This will be our maximum value. I am getting square root of given number to increase the performance of this method by decreasing number of iterations in the for loop.
4. In step I am performing whether given number is even number or not. In this step I have a for loop and this for loop will generate odd numbers and those numbers will be my divisor and given number will be divided by those generated odd numbers and check whether we have a remainder or not. If we do not have a remainder that means given number is divisible by for loop generated odd number. So given number is not a prime number and it is divisible by another number, then I return false. If given number cannot divide by odd number without having any remainder that means given number is a prime number.

### Flow chart of is\_prime\_method



## Generate\_prime\_number method

This function is not accepting any argument but return a prime number. This method has a never ending while loop, it will iterate until it find a prime number.

Following steps describe how algorithm works to generate a prime number.

1. As a first step I generate random number which will be between 10 and maximum number we created outside the class.
2. In the second step I call is\_prime method inside if condition by passing randomly generated number to check whether that number is a prime number or not. If our number is prime number is\_prime number will return true else return false. If it is a prime number, return that number and terminate never ending while loop. If our number is not a prime number while loop will iterate until it finds a prime number.



## Generate\_keyPairs method

This function accepts two prime numbers and return tupple of private and public key pair. Following methods explain how this function works.

1. As a first step I am calculating n value by multiplying two prime numbers.
2. In the second step I am calculating and assign phi value to a variable.
3. In third step I am generating random value between 1 and calculated phi value. Which will be assigned to a variable.
4. As forth step I am calculating greatest common divisor of e and phi by using math library in the python and assigned to a variable.
5. In fifth step I am checking calculated common divisor is equal to one or not inside the while loop. If greatest common divisor is not equal to one, then while loop will execute and again generate random number for e and calculate greatest common divisor. This process will continue until common divisor is equal to one.
6. In next step I am calling imod function to get modular inverse of calculated e and phi.
7. In the next step I am checking calculated modular inverse value is a negative or not. If it is a negative number I am converting that to positive number.
8. As a last step I am returning a tupple with private key and public key.

## Pow\_function method

This method is the most important method in this project. This method is accepting three different inputs. Those are base, exponent and another number. Main purpose of this function is to calculate a power of number by using given base number and exponent number and calculate modulus. This will return a number after completing calculation. I am using bitwise operations inside this method to increase performance of my application.

To get xy % n we have to multiply x number y times and have to calculate modulus. When we multiply several times one number, we get a big value, that big value might exceed the maximum value can integer handle. To handle big numbers, I am calculation modulus operation after every multiplication. Every multiplication reduces the number of iterations by half. I am doing the right bit shift to divide the number of iterations by two. I am doing right bit shift until number of iterations become zero. After number of iterations become zero, program get exit from the never-ending loop.

### Flow chart of pow\_function method



## Encrypt method

This method is responsible of encrypting given text by using generated public key. Mathematical equation for encryption is c=me (mod n). I am splitting each character in the given text message and convert each character into Unicode. After converting into Unicode I am passing that value into pow\_function with public key. Which performs me (mod n) operation and return a value. ”m” is my Unicode and e and n are my public key. I am appending return value of pow\_function into a list. That list is my cypher text.

## Decrypt method

This method is responsible of decrypting a cypher text using generated private key. Mathematical equation for decryption is cd (mod n). Our cypher text is a list. I extract each element from the list and using pow\_function I will perform cd (mod n) operation. “c” is element split from the cipher list. ”d” and “n” is my private key. When we perform cd (mod n) function we get Unicode value. Next, I am converting that Unicode value back to original text or number. After conversion I am appending as a string into another list. At the end I am merging all elements in the list and return original message.

# Model.py file

## Logic class

Logic class contains all methods associated with button click events. In the constructor of Logic class, I am creating Rsa object and Interface object.

### Clear\_all\_fields method

Main functionality of this method is to delete contents in the entry fields.

### Button\_click\_encrypt method

This function get trigger when user click the Encrypt button. Main functionality of this method is to validate user input and call key generation, encryption and decryption methods and assign values to relevant entry fields.

### Button\_click\_generate\_keys method

This method is responsible of calling generate\_keypairs method and get public and private key. This function is doing validation of user inserted prime numbers before calling key generating method.

### Clear\_keys method

I am maintaining private key and public key as instance variables. Main functionality of this method is to clear values in the public and private key instance variables and delete private and public key entry fields in the user interface.

### Clear\_cipher method

Main responsibility of this function is to remove values in the cipher instance variable and cipher entry field.

### Check\_user\_inserted\_prime method

This method reads user inserted prime numbers from the entry fields and validate whether they are integers or not. If they are integers again check whether they are prime numbers or not. If user insert string this function generate warning message and clear that entry field. If entry fields are empty this method is calling generate prime numbers method and get prime numbers and put those prime number into relevant entry fields.

# Performance Enhancement

To improve the performance of my application I am using bitwise operations inside pow\_function. Moreover, I am using another technique to minimize time taken to perform Xy %d kind of operations. When performing Xy %d kind of operations I do not multiply X, Y times and perform modulus. I am multiplying X with X and perform modulus. After that operation I am doing right bit shift of Y and get half value of Y. Y represent number of iterations. In every iteration I am performing right bit shift of the Y to get half value of Y. I have enhanced the performance of is\_prime method as well. I do not loop given number times to find whether given number is a prime number or not. I am calculating square root of the given number and that square root is my number of iterations.

# Weaknesses

This application is not accepting prime numbers less than 10. Because decryption is not working properly when use prime numbers less than 10. This is a limitation of my application. Second limitation of my application is I have limit in the range of prime numbers. I am calculating maximum number system can handle and get square root of that number, and I use that number as maximum number to generate random prime numbers.