CS4003701 資訊安全導論 Introduction to Information Security
HW4 Report

B10832008 蔡芸軒

[Structure tree]

```
github : 110-Information-Security
| ----HW1
| ----HW2
| ----HW3
      | ---- RSA.py
      | ---- Report.pdf
```

[key generation]

First, I generate two n-bits random numbers P and Q, and then use the **Miller Rabin test** to check if it's prime. If not, regenerate it until it's prime.

[Miller Rabin test]

Given a number `n`, randomly select a number a such that `1 < a < n`.
If we can find an `a` that `(a^d)%n != 1` and `(a^(2^r*d))%n != -1`, then `n` is said to be a strong probable prime to base `a`, `n` isn't prime.
Else, `a` is a strong liar that may prove `n` is prime.

[encrypt]

calculate `(message ^ e) % e`
and accelerate by **square and multiply** method instead of simply **pow**

[decrypt]

calculate `(message ^ e) % d`
and accelerate by **square and multiply** method instead of simply **pow**

[pow by square and multiply]

Starting with the least significant bit of exponent, progressively square the base at each step to get factor.
And if the bit is 1, multiply the running product by new factor.

[CRA]

do encrypt and decrypt, and accelerate by chinese remainder theorem