RSA Encryption/Decryption

Introduction

Development team members:

Weiheng Ruan(Alex): waruan2@uic.edu

Byambasuren Gansukh(Ben): bgansu3@uic.edu

Section 1 – Purpose

The purpose of RSA Encryption/Decryption is to allow user to create a public and private key which are use to encrypting and decrypting messages. To run the program the user must have Java install. The program will communicate with the user on a GUI platform.

Section 2 – High level entities

In this project there will exist 3 class(5 if use inheritance for encryption and decryption)
The classes include: Huge-Unsigned-Integer Class, RSA, UI Class or classes depending on how much we'd want to modularize it.

Section 3 – For each entity, define the low level design:

3.1 Huge-Unsigned-Integer (HUI) Class

The Huge Unsigned Integer (HUI) will be a replacement for the Integer class. The HUI will be constructed using a dynamic array of either integer.The value will be store from less significant to most significant. The HUI class will have the function of addition, subtraction, multiplication, division, modulus, and relational operations

3.1.1 Addition

The addition function will take either two Huge-Unsigned-Integer and return the sum of the two HUI as a HUI or it will take an Integer and an HUI and return the sum of the integer and the HUI as a HUI.

3.1.2 Subtraction

The subtraction function will take either two Huge-Unsigned-Integer and return the difference of the two HUI as a HUI or it will take an Integer and an HUI and return the difference of the integer and the HUI as a HUI.

3.1.3 Multiplication

The multiplication function will take either two Huge-Unsigned-Integer and return the product of the two HUI as a HUI or it will take an Integer and an HUI and return the product of the integer and the HUI as a HUI.

3.1.4 Division

The division function will take either two Huge-Unsigned-Integer and return the quotient of the two HUI as a HUI or it will take an Integer and an HUI and return the quotient of the integer and the HUI as a HUI

3.1.5 Relational Operations (Equal)

The equal function will take two HUI and return true if the two HUI are same or false if the two are different

3.1.6 Relation Operation (Greater Than)

The greater function will compare the first HUI to the second HUI. If the first HUI is greater the function will return true else it will return false.

3.1.7 Relational Operations (Greater or Equal)

The greater_or_equal function will take two HUI as parameter. This function will first make a call to the equal function to compare to see if the HUI are equal, if the equal function return true the greater_or_equal return true. Else it will call the greater function and return the value it receive from the greater function.

3.18 Relation Operation (Less Than)

The less function will compare the first HUI to the second HUI. If the first HUI is smaller the function will return true else it will return false.

3.1.9 Relational Operations (Less Than or Equal)

The less_or_equal function will take two HUI as parameter. This function will first make a call to the equal function to compare to see if the HUI are equal, if the equal function return true the less_or_equal return true. Else it will call the lessthan function and return the value it receive from the lessthan function.

3.2 UI

The UI will allow the user to choose one these four option.

1. Key Creation – Create the private key file and public key file from the two prime numbers of p and q.
2. Block a file – Create a “blocked file” from an ASCII text file and a block size value
3. Unblock a file – Create an ASCII text file from a “blocked file”.

4. Encrypt/Decrypt – Create a new “blocked file” that is the result of running the encryption algorithm on each line of another blocked file using the key pair contained in a file specified by the user

3.3 ViewFile - Static class that deals with opening a user specified file in a non writable format in a window or dialog message pane. Methods of this class are all static. Strongly tied with the GUI.

- 3.3.1 - PromptUserForFile() - Prompt the user to select a some file to view.
- 3.3.2 - void OpenFile(File filePath) - Open the user specified file in a non writable format.

3.4 KeyCreation - Class that deals with creating a both public and private keys.

- 3.4.1 - KeyCreation(HUI prime1, HUI prime2) - Constructor that takes 2 prime numbers and also creates the Both keys.
- 3.4.2 - PublicKey GetPublicKey() - Getter that returns the PublicKey pbject.
- 3.4.3 - PrivateKey GetPrivateKey() - Getter that returns the PrivateKey object.
- 3.4.4 - void createD() - create d assuming all the needed values are initialized.
  $d = (1 + k\varphi)/e \rightarrow$ *for some integer* k, d *is the quotient of* $(1 + k\varphi)/e$ *when* $(1 + k\varphi)$ *can be evenly divided by* e .
- 3.4.5 - void createE() - create e assuming all the needed values are initialized.
  e is some arbitrary number that is less than n and relatively prime to $\varphi = (prime1 - 1) * (prime2 - 1)$
- 3.4.6 - void createN() - create n assuming all the needed values are initialized.
  $n = prime1 * prime2$

3.5 Key - Class that deals with storing a key. A key consists of two values.

- 3.5.1 - Key(HUI value1, HUI value2) - initializes the data members of Key.
- 3.5.2 - HUI getN() - Getter that returns the value of n. Both public and private keys has a n field which makes this convenient.

3.6 PublicKey extends Key - class that extends Key, provides getters.

- 3.6.1 - PublicKey(HUI e, HUI n) - Calls super(e, n).
- 3.6.2 - HUI getE() - Getter that returns the value of e.

3.7 PrivateKey extends Key - class that extends Key, provides getters.

- 3.7.1 - PrivateKey(HUI d, HUI n) - Calls super(d, n).
- 3.7.2 - HUI getD() - Getter that returns that value of d.

3.8 Block - Class that deals with blocking a file

- 3.8.1 - Block(File filePath) - blocks a file with a fixed block size of 10 and store it in someDataStructure.
- 3.8.2 - Block(someDataStructure fileData) - blocks the data same as the above constructor.
- 3.8.3 - someDataStructure getBlockedData() - returns the blocked file in a format of someDataStructure.
- 3.8.4 - void BlockFile() - module that does the blocking.

3.9 Unblock - Class that deals with unblocking a file

- 3.9.1 - Unblock(File filePath) - unblocks a file and store it in someDataStructure.
- 3.9.2 - Unblock(someDataStructure fileData) - unblocks a file.
- 3.9.3 - someDataStructure getUnblockedData() - returns the unblocked file in a format of someDataStructure.
- 3.9.4 - void UnblockFile() - module that does the unblocking.

3.10 Encrypt - Class that deals with encrypting a file

- 3.10.1 - Encrypt(PublicKey key, File filePath) - takes in a PublicKey and a file to encrypt then creates Block(filePath) to block the file and encrypts it. Stores the encrypted file in someDataStructure to be stored in a file.
- 3.10.2 - void encrypt() - module that does the encrypting and storing.
- 3.10.3 - someDataStructure getEncryptedData() - returns the encrypted data in a format of someDataStructure.

3.11 Decrypt - Class that deals with decrypting a file

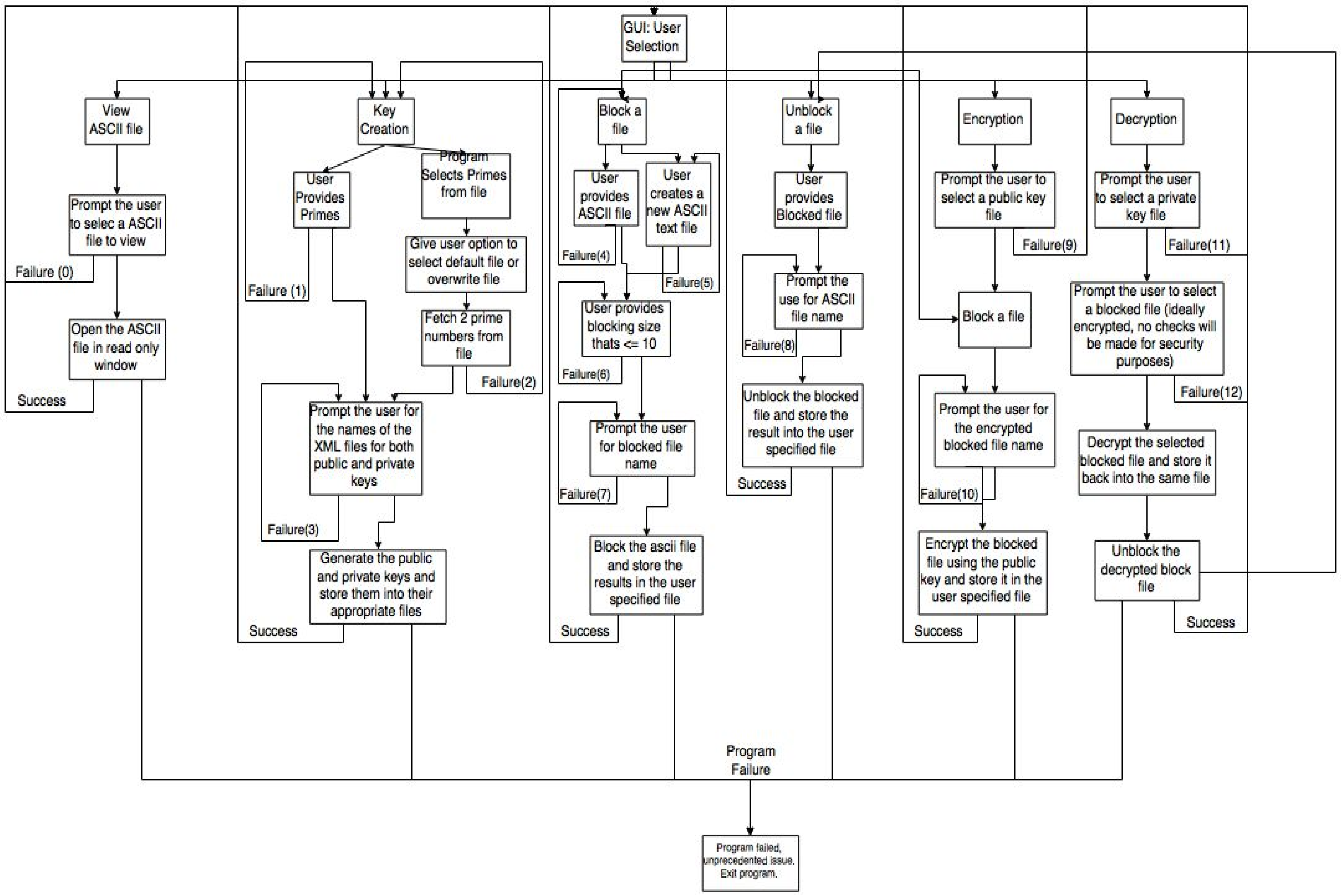
- 3.11.1 - Decrypt(PrivateKey key, File filePath) - takes in a PrivateKey and a file and decrypts it. Stores the decrypted file in someDataStrcture then creates Unblock(someDataStructure) to unblock the decrypted data.
- 3.11.2 - void decrypt() - module that actually does the decrypting.
- 3.11.3 - someDataStructure getDecryptedData() - returns the unblocked data in a format of someDataStructure.

3.12 RSAHelper - Helper static class that deals with checking for validity on few entities.

- 3.12.1 - boolean isPrimeNumber(HUI prime1) - returns true if prime1 is a prime number. Else returns false.
- 3.12.2 - boolean GCD1(HUI e, HUI x) - returns true if e is relatively prime to x. Else return false.

Section 4 – Program Flow:

This is just an overview of how the GUI and the program works.


GUI: User Selection
View ASCII file
Prompt the user to selec a ASCII file to view
Failure (0)
Open the ASCII file in read only window
Success
Key Creation
User Provides Primes
Program Selects Primes from file
Give user option to select default file or overwrite file
Fetch 2 prime numbers from file
Failure (1)
Failure(2)
Prompt the user for the names of the XML files for both public and private keys
Failure(3)
Generate the public and private keys and store them into their appropriate files
Success
Block a file
User provides ASCII file
User creates a new ASCII text file
Failure(4)
Failure(5)
User provides blocking size thats <= 10
Failure(6)
Prompt the user for blocked file name
Failure(7)
Block the ascii file and store the results in the user specified file
Success
Unblock a file
User provides Blocked file
Prompt the use for ASCII file name
Failure(8)
Unblock the blocked file and store the result into the user specified file
Success
Encryption
Prompt the user to select a public key file
Failure(9)
Block a file
Prompt the user for the encrypted blocked file name
Failure(10)
Encrypt the blocked file using the public key and store it in the user specified file
Success
Decryption
Prompt the user to select a private key file
Failure(11)
Prompt the user to select a blocked file (ideally encrypted, no checks will be made for security purposes)
Failure(12)
Decrypt the selected blocked file and store it back into the same file
Unblock the decrypted block file
Success
Program Failure
Program failed, unprecedented issue. Exit program.

Failure Descriptions:

Program - Failure	Any unprecedented error that might restrict a task from being successfully executed.
Failure(0) -	There aren't any ASCII files to choose from or the user cancels request. Program flow restarts at GUI: User Selection.
Failure(1) -	User provides non prime number. Program flow restarts at Key Creation.

Failure(2) -	There aren't prime numbers in file or the file doesn't exist. Program flow restarts at Key Creation.
Failure(3) -	User provides invalid filenames(s). Reprompt the user for valid names.
Failure(4) -	No ASCII file exists in the directory. Program flow restarts at Block a file.
Failure(5) -	User tries to save the new ASCII file with a name that already exists. Reprompt for a valid name.
Failure(6) -	User provides an invalid blocking size, type or value. Reprompt the user for a valid blocking size.
Failure(7) -	User provides an invalid file name. Reprompt the user for a valid filename.
Failure(8) -	User provides an invalid file name. Reprompt the user for a valid filename.
Failure(9) -	Unable to select a public key. Program flow starts at GUI: user selection where user can choose to create a key.
Failure(10) -	User provides an invalid file name. Reprompt the user for a valid filename.
Failure(11) -	Unable to select a private key. Program flow starts at GUI: user selection where user can choose to create a key.
Failure(12) -	Unable to select a blocked file name. Program flow starts at GUI: user selection.

Section 5 – Benefits, assumptions, risks/issues:

Benefit:

1) By allowing function overloading for algebra operation function. It will reduce the amount of memory needed. Also it allow more freedom for the user of the HUI class because the user will not be restricted to creating a new HUI before doing the algebra operation.
2) The encryption and decryption class does not depend on the HUI class. The two class can use regular integer and it will still function correct
3) Allow user to input file and instead of manually inputting information every time
4) Output of the encryption and decryption will either be display on the GUI or output to a file

Assumption / Risk

1) The primal number the user input will be large enough for creating a private and public key
2) The algebra operation result will not larger than the memory can handle
3) If a negative number return from HUI algebra operation the program will throw an exception and exit
4) Assuming user will only input “number” String such that the string only contain numbers