David San

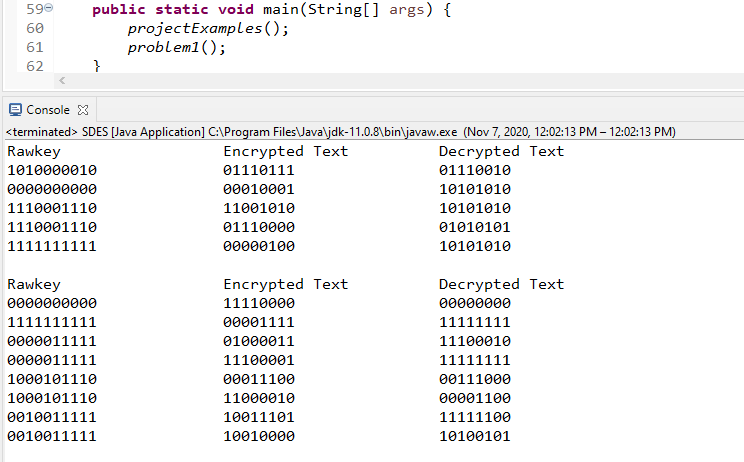
Aldo Gil

CS 4780-01

Project 1

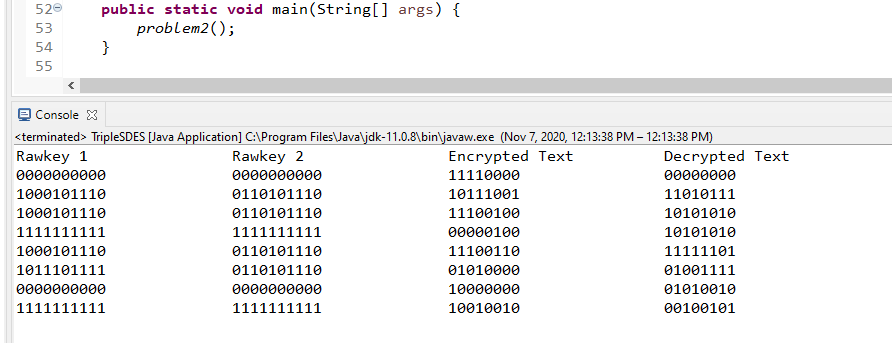
Due Nov 7

In order to run the project, make sure all files (CASCII.java, Cracking.java, SDES.java, and TripleSDES.java) are in the same package in your IDE. Each problem is separated in its own java file, but Cracking.java utilizes some other files, so all java files are needed in order to work correctly.

1. **SDES**  
   To run SDES, open the SDES file in the IDE and run the file. In the console, it should print out the examples we were given for the project, as well as the questions we were given to solve (as shown below in pic 1). In order to run the Encryption/Decryption by itself, you would just need to run the appropriate function (i.e. Encrypt(rawkey, plaintext); ) and it will return an array of bytes. Our class turns those byte arrays to strings and prints it out so the user can see the result in a more appealing way.  
     
   (Pic 1 - SDES.java running and showing the results of the examples and questions from the project)

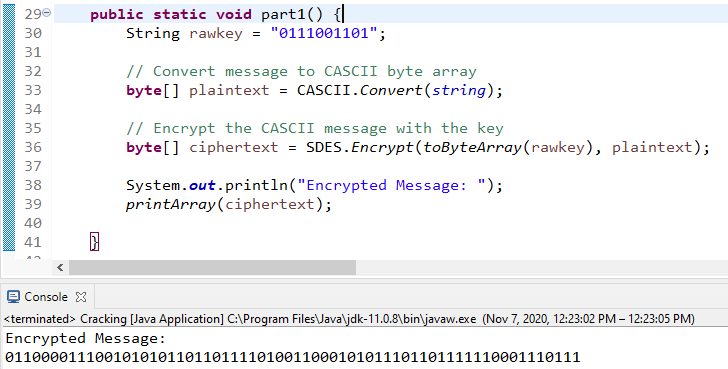
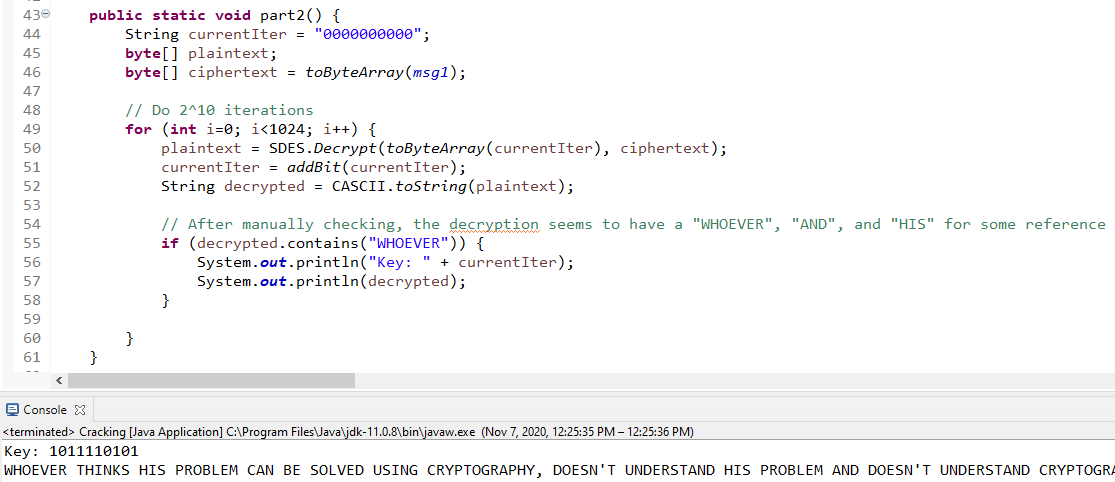
|  |  |  |
| --- | --- | --- |
| Rawkey | Plaintext | Cipherkey |
| 00000 00000 | 0000 0000 | 1111 0000 |
| 11111 11111 | 1111 1111 | 0000 1111 |
| 00000 11111 | 0000 0000 | 0100 0011 |
| 00000 11111 | 1111 1111 | 1110 0001 |
| 10001 01110 | 0011 1000 | 0001 1100 |
| 10001 01110 | 0000 1100 | 1100 0010 |
| 00100 11111 | 1111 1100 | 1001 1101 |
| 00100 11111 | 1010 0101 | 1001 0000 |

(Table 1 - Results for the problems given in the project highlighted in green)  
Also attached is a table with the results for the problems given in the project.

1. **TripleSDES**  
   TripleSDES works just like SDES. Open the file and run the file in order to see the results. The only real difference for TripleSDES was that we had to have 2 keys, and run the specific algorithm - Encrypt(Decrypt(Encrypt) ) or vice versa in order to obtain the results we wanted. So in this class, we added a EncryptMini and DecrpytMini which are the same methods as Encrypt and Decrypt from SDES so we can run the triple in the main methods for this class.  
     
   (Pic 2 - TripleSDES.java running displaying the results of the problems from the project)

|  |  |  |  |
| --- | --- | --- | --- |
| Raw Key 1 | Raw Key 2 | Plaintext | Ciphertext |
| 00000 00000 | 00000 00000 | 0000 0000 | 1111 0000 |
| 10001 01110 | 01101 01110 | 1101 0111 | 1011 1001 |
| 10001 01110 | 01101 01110 | 1010 1010 | 1110 0100 |
| 11111 11111 | 11111 11111 | 1010 1010 | 0000 0100 |
| 10001 01110 | 01101 01110 | 1111 1101 | 1110 0110 |
| 10111 01111 | 01101 01110 | 0100 1111 | 0101 0000 |
| 00000 00000 | 00000 00000 | 0101 0010 | 1000 0000 |
| 11111 11111 | 11111 11111 | 0010 0101 | 1001 0010 |

(Table 2 - Results for the problems given in the project highlighted in green)

1. **Cracking SDES and Triple SDES**The Cracking java file contains 3 parts from the project and are separated into 3 methods in the class file. As a reminder, this class utilizes other java files, so having all java files are needed for this class to properly work.
   1. **Encode the CASCII plaintext CRYPTOGRAPHY using SDES, key of 01110 01101.**  
      For this problem, we converted the “CRYPTOGRAPHY” into a byte array using the CASCII class, then encrypted the text using our SDES class.  
        
      (Pic 3-1 - Conversion and encryption for the message and the encrypted message result)
   2. **Decrypt msg1.txt using SDES and provide the 10-bit key**  
        
      For this problem, we ran the function multiple times to make sure it was right; and we ended by making it simple for the user to see the end result by manually finding a key phrase for the decrypted message. **The final message ended up being “WHOEVER THINKS HIS PROBLEM CAN BE SOLVED USING CRYPTOGRAPHY, DOESN'T UNDERSTAND HIS PROBLEM AND DOESN'T UNDERSTAND CRYPTOGRAPHY. ATTRIBUTED BY ROGER NEEDHAM AND BUTLER LAMPSON TO EACH OTHER” from a key of 10111 10101.**  
      (Pic 3-2 - SDES Decryption of the message and the result.  
        
      
   3. **Decrypt msg2.txt using TripleSDES and provide the 2 10-bit keys**  
        
      This problem was tackled the same way as the one with SDES; we just had to wait longer and manually search more since there were 2 keys and a lot more text to go through. This one, we also made a File reader to export the text so that it would be easier to search through the results. **The final message ended up being “THERE ARE NO SECRETS BETTER KEPT THAN THE SECRETS THAT EVERYBODY GUESSES.” with key1 as 11100 00101 and key2 as 01011 00011.**(Pic 3-3 - TripleSDES decoding of the message and the result)