Advanced Security 1 – Lab Sheet 2 – Stephen Darcy C18490924

1. Write a program that will implement Caesar Cipher and Vigeneré Cipher. You can use Java or any other programming language.
2. *package* com.lab2.cipher;  
     
   */\*  
   Author: Stephen Darcy  
   Date: 04/10/2021  
   Description: Write a program that will implement Caesar Cipher and Vigenère Cipher. You can use  
   Java or any other programming language.  
   \*/  
     
   import* java.util.Arrays;  
   *import* java.util.Locale;  
   *import* java.util.Scanner;  
     
   */\*\*  
    \* Console program that allows the user to encrypt and decrypt two ciphers; Caesar cipher and Vigenère cipher  
    \*/  
     
   public class* Cipher {  
    *static* Scanner scanner = *new* Scanner(System.in);  
     
    *public static void* main(String[] args) {  
    *int* menuOption = 0;  
     
    *//wrap menu in while statement so it loops  
    while* (menuOption != 5) {  
    *//getting menu option from user* menuOption = menu();  
    scanner.nextLine();  
    *//switch based on users input  
    switch* (menuOption) {  
    *case* 1 -> encryptCaesar();  
    *case* 2 -> Vigenere(*true*);  
    *case* 3 -> decryptCaesar();  
    *case* 4 -> Vigenere(*false*);  
    *case* 5 -> System.out.println("\tExiting...");  
    *default* -> System.out.println("\n\tError invalid input, returning to menu ...");  
    }  
    }  
     
     
    }  
     
     
    */\*\*  
    \* Asks the user what their ciphertext has been shifted by. Then gets the user to enter their ciphertext  
    \* before printing the decrypted plaintext  
    \*/  
    private static void* decryptCaesar() {  
    System.out.println("\t----- Caesar Cipher Decryption Selected -----");  
    *//asking the user what the shift was for the ciphertext* System.out.print("\tPlease enter the number your ciphertext was shifted by (Caesar cipher is traditionally 3):\n\t");  
    *int* shift = getInt();  
    scanner.nextLine();  
     
    *//getting the ciphertext from the user before decryption* System.out.print("\tPlease enter the ciphertext to be decrypted:\n\t");  
    String ciphertext = scanner.nextLine();  
    ciphertext = ciphertext.toLowerCase();  
    ciphertext = ciphertext.replace("\"", "");  
    System.out.println("\tYou entered: " + ciphertext);  
     
    *//putting ciphertext into char array & adding a plaintext stringbuilder  
    char*[] ciphertextArray = ciphertext.toCharArray();  
    StringBuilder plaintext = *new* StringBuilder();  
     
     
    *//decrypting the ciphertext  
    for* (*int* i = 0; i < ciphertext.length(); i++) {  
    *char* currentLetter = ciphertextArray[i];  
    *if* (currentLetter == ' ') {  
    *continue*;  
    }  
    *//making sure plaintextArray[i] is a char; if not setting as a space ' '  
    if* (Character.isLetter(ciphertextArray[i])) {  
    *//using ASCII values to find new shift position in ASCII  
    int* offset = ((ciphertextArray[i] - 'a') - shift) % 26;  
     
    *//converting ASCII value and adding new values to ciphertext* plaintext.append((*char*) (offset + 'a'));  
    }  
    }  
    *//printing the ciphertext to the user* System.out.println("\tPlaintext is: " + plaintext);  
    }  
     
     
    */\*\*  
    \* Asks the user to input their desired text and keyword. The keyword is converted to a key and the key  
    \* is then used to encrypt the plaintext. If the user chooses decryption the key is subtracted from the text instead  
    \* key and plaintext are both made uppercase.  
    \*/  
    private static void* Vigenere(*boolean* option) {  
    String choice;  
     
    *if* (option) {  
    choice = "Encrytion";  
     
    } *else* {  
    choice = "Decrption";  
    }  
     
    System.out.println("\t----- Vigenere Cipher " + choice + " Selected -----");  
    *//getting the users text* System.out.print("\tPlease enter the text to begin " + choice + " :\n\t");  
    String text = scanner.nextLine();  
    text = text.toUpperCase();  
    System.out.println("\tYou entered: " + text);  
     
    *//putting text into char array  
    char*[] textArray = text.toCharArray();  
     
    *//getting the users keyword* System.out.print("\tPlease enter the keyword:\n\t");  
    String keyword = scanner.nextLine();  
    keyword = keyword.toUpperCase();  
    System.out.println("\tYou entered: " + keyword);  
     
    *//creating a key array the length of plaintext  
    char*[] key = *new char*[text.length()];  
     
    *//converting the keyword into a key (repeat keyword until key.length == text.length)  
    int* i, j;  
    *for* (i = 0, j = 0; i < text.length(); i++, j++) {  
    *//resetting j when it reaches the length of keyword  
    if* (j == keyword.length()) {  
    j = 0;  
    }  
    *//making the key* key[i] = keyword.charAt(j);  
    }  
     
    *//char array to hold the final text  
    char*[] text1 = *new char*[text.length()];  
    *if* (option) {  
    *//encrypting the message with the key  
    for* (i = 0; i < text.length(); i++) {  
    text1[i] = (*char*) (((textArray[i] + key[i]) % 26) + 'A');  
    }  
    } *else* {  
    *//decrypting the message with the key  
    for* (i = 0; i < text.length(); i++) {  
    text1[i] = (*char*) ((((textArray[i] - key[i]) + 26) % 26) + 'A');  
    }  
    }  
     
     
    *//displaying the text* System.out.println("\tText is: " + *new* String(text1));  
     
     
    }  
     
    */\*\*  
    \* Asks the user to input their desired shift. User is then prompted to enter plaintext  
    \* which in turn returns ciphertext  
    \*/  
    private static void* encryptCaesar() {  
    System.out.println("\t----- Caesar Cipher Encryption Selected -----");  
    *//getting the users desired shift* System.out.print("\tPlease enter the number you wish to shift your plaintext by (Caesar cipher is traditionally 3):\n\t");  
    *int* shift = getInt();  
    scanner.nextLine();  
     
    *//getting the users plaintext* System.out.print("\tPlease enter the plaintext to be encrypted:\n\t");  
    String plaintext = scanner.nextLine();  
    plaintext = plaintext.toUpperCase();  
    System.out.println("\tYou entered: " + plaintext);  
     
    *//putting plaintext into char array & adding ciphertext stringbuilder  
    char*[] plaintextArray = plaintext.toCharArray();  
    StringBuilder ciphertext = *new* StringBuilder();  
     
    *//encrypting the plaintext  
    for* (*int* i = 0; i < plaintext.length(); i++) {  
    *//making sure plaintextArray[i] is a char; if not setting as a space ' '  
    if* (Character.isLetter(plaintextArray[i])) {  
    *//using ASCII values to find new shift position in ASCII  
    int* offset = ((plaintextArray[i] - 'A') + shift) % 26;  
     
    *//converting ASCII value and adding new values to ciphertext* ciphertext.append((*char*) (offset + 'A'));  
    } *else* {  
    ciphertext.append(" ");  
    }  
    }  
     
    *//printing the ciphertext to the user* System.out.println("\tCiphertext is: " + ciphertext);  
     
    }  
     
    */\*\*  
    \* Print a menu and return the users input  
    \*  
    \* @return the menu option the user selected  
    \*/  
    public static int* menu() {  
    System.out.println("\n\tLab 2 program Caesar cipher and Vigenère cipher\n");  
    System.out.println("\tPlease choose an option below: \n -----------------------------------------------");  
    System.out.println("\t1. Encrypt with Caesar cipher");  
    System.out.println("\t2. Encrypt with Vigenère cipher");  
    System.out.println("\t3. Decrypt a Caesar cipher");  
    System.out.println("\t4. Decrypt a Vigenère cipher");  
    System.out.println("\t5. Quit\n");  
    System.out.print("\t");  
     
    *return* getInt();  
    }  
     
    */\*\*  
    \* prompts the user to enter an int, stopping the program from crashing when another datatype  
    \* is entered.  
    \*  
    \* @return an int the user inputted  
    \*/  
    public static int* getInt() {  
    *//waiting for user to enter an integer  
    while* (!scanner.hasNextInt()) {  
    scanner.next();  
    System.out.print("\tPlease enter an integer value\n\t");  
    }  
     
    *return* scanner.nextInt();  
    }  
   }

3.

Onevariationtothestandardcaesarcipheriswhenthealphabetiskeyedbyusingawordinthetraditionalvarietyonecouldwritethealphabetontwostripsandjustmatchupthestripsafterslidingthebottomstriptotheleftorrighttoencodeyouwouldfindaletterinthetoprowandsubstituteitfortheletterinthebottomrowforakeyedversiononewouldnotuseastandardalphabetbutwouldfirstwriteawordomittingduplicatedlettersandthenwritetheremaininglettersofthealphabetfortheexamplebelowiusedakeyofrumkincomandyouwillseethattheperiodisremovedbecauseitisnotaletteryouwillalsonoticethesecondmisnotincludedbecausetherewasanmalreadyandyoucanthaveduplicates

1. key = 17
2. NIST IS ABOUT TO ANNOUNCE THE NEW HASH ALGORITHM THAT WILL  
   BECOME SHA-3. THIS IS THE RESULT OF A SIX-YEAR COMPETITION, AND  
   MY OWN SKEIN IS ONE OF THE FIVE REMAINING FINALISTS (OUT OF  
   AN INITIAL 64). IT'S PROBABLY TOO LATE FOR ME TO AFFECT THE FINAL  
   DECISION, BUT I AM HOPING FOR "NO AWARD." IT'S NOT THAT THE NEW  
   HASH FUNCTIONS AREN'T ANY GOOD, IT'S THAT WE DON'T REALLY NEED  
   ONE. WHEN WE STARTED THIS PROCESS BACK IN 2006, IT LOOKED AS IF  
   WE WOULD BE NEEDING A NEW HASH FUNCTION SOON. THE SHA  
   FAMILY (WHICH IS REALLY PART OF THE MD4 AND MD5 FAMILY), WAS  
   UNDER INCREASING PRESSURE FROM NEW TYPES OF  
   CRYPTANALYSIS. WE DIDN'T KNOW HOW LONG THE VARIOUS SHA-2  
   VARIANTS WOULD REMAIN SECURE. BUT IT'S 2012, AND SHA-512 IS STILL  
   LOOKING GOOD.  
   EVEN WORSE, NONE OF THE SHA-3 CANDIDATES IS SIGNIFICANTLY  
   BETTER. SOME ARE FASTER, BUT NOT ORDERS OF MAGNITUDE FASTER.  
   SOME ARE SMALLER IN HARDWARE, BUT NOT ORDERS OF MAGNITUDE  
   SMALLER. WHEN SHA-3 IS ANNOUNCED, I'M GOING TO RECOMMEND  
   THAT, UNLESS THE IMPROVEMENTS ARE CRITICAL TO THEIR  
   APPLICATION, PEOPLE STICK WITH THE TRIED AND TRUE SHA-512.  
   AT LEAST FOR A WHILE. I DON'T THINK NIST IS GOING TO ANNOUNCE  
   "NO AWARD"; I THINK IT'S GOING TO PICK ONE. AND OF THE FIVE  
   REMAINING, I DON'T REALLY HAVE A FAVORITE. OF COURSE I WANT  
   SKEIN TO WIN, BUT THAT'S OUT OF PERSONAL PRIDE, NOT FOR SOME  
   OBJECTIVE REASON. AND WHILE I LIKE SOME MORE THAN OTHERS, I  
   THINK ANY WOULD BE OKAY. WELL, MAYBE THERE'S ONE REASON NIST  
   SHOULD CHOOSE SKEIN. SKEIN ISN'T JUST A HASH FUNCTION, IT'S  
   THE LARGE-BLOCK CIPHER THREEFISH AND A MECHANISM TO TURN IT  
   INTO A HASH FUNCTION. I THINK THE WORLD ACTUALLY NEEDS A  
   LARGE-BLOCK CIPHER, AND IF NIST CHOOSES SKEIN, WE'LL GET ONE.
3. Swahili

chama cha mapinduzi pamoja na vijana wake kupitia umoja wao wa uvccm,

kimemshukia aliyekuwa mwenyekiti wa tume ya mabadiliko ya katiba, jaji joseph

warioba, kikimtaka aache kujidanganya, kwani suala la katiba mpya haliwezi kuwa

ajenda ya uchaguzi mkuu, mwakani. kwa upande wa uvccm, imemtaka jaji warioba,

aache mara moja kutumia dhamana aliyokuwa amepewa ya kuwa mwenyekiti wa tume ya

mabadiliko ya katiba, kwani muda wake umeishamalizika kisheria. kauli hizo

zilitolewa kwa nyakati tofauti na viongozi wa chama hicho, ikiwa ni siku chache tangu

jaji warioba atoe maoni yake kuhusiana na rasimu iliyopendekezwa na bunge maalum

la katiba, ambapo alikosoa kutokana na kuachwa kwa baadhi ya maoni ya wananchi.

aidha, ameendelea kusisitiza kuwa, atakuwa rais wa watanzania, bila kujali dini,

kabila au vyama, hivyo maendeleo ya serikali yake hayatabagua. akizungumza jana

mjini hapa kwenye mkutano wa kampeni uliohudhuriwa na maelfu ya watu ambao alikiri

kuwa ni mkubwa ambao hajawahi kuuona, amewahakikishia kuwa ataiendesha nchi kwa

ustaarabu na si kwa udikteta kama ambavyo baadhi ya watu wamekuwa wakidai.

hata baada ya kuchaguliwa, mimi sitabadilika, nitabaki kuwa mtoto wenu yule yule

john magufuli, alisema na kuongeza; nitaiendesha nchi kwa ustaarabu, sitaiendesha

nchi kwa udikteta pamekuwa na watu wanazungumza, kwa sababu nazungumza ukweli na

ukweli utabaki ukweli kweli. watu wanabaki kutishiana. nyie wana chato waelezeni

ukweli kwamba nilipokuwa waziri nilikuwa nachunga ng’ombe, nilikuwa nakamua maziwa.