Introducción a las Ciencias de la Computación
2020-2
Proyecto 2

Pedro Ulises Cervantes González
confundeme@ciencias.unam.mx

Adriana Sánchez del Moral
adrisanchez@ciencias.unam.mx

Emmanuel Cruz Hernández
emmanuel_cruzh@ciencias.unam.mx

Fecha de entrega: 19 de abril de 2020

26 de marzo de 2020

Objetivo

Programar tres tipos distintos de cifrado de mensajes.

Entrada

Texto arbitrario que deben leer. Lo pueden leer de consola para probar su clase, pero tienen que redirigir la entrada desde un archivo en texto para entregar.

Salida

El texto original, el texto cifrado y el texto descifrado en un archivo en disco.

Observación

Se eliminan del texto original todos los blancos, caracteres especiales y dígitos.

Método

En todos los casos deberán tener los métodos de acceso para los atributos, que son, fundamentalmente, el mensaje sin cifrar y el mensaje cifrado.

Notas

- Los caracteres se tienen que convertir explícitamente usando ***Character.toString(char c)*** que es un método estático de la clase ***Character***, que es un *wrapper* del tipo primitivo **char**.
- Tienen que redirigir tanto la entrada como la salida en la invocación del programa.
- Tienen que montar al ***Scanner*** sobre ***System.in***

Cifrado de César

El cifrado de César, también conocido como cifrado de desplazamiento, es una de las técnicas más simples de cifrado. Consiste en reemplazar cada caracter del alfabeto por el que se encuentra k posiciones a la derecha en el mismo alfabeto. Consideramos al alfabeto circular, esto es, sacamos módulo 26 a la posición que nos dé. Si $e(x)$ me da la posición en el alfabeto del caracter cifrado y x me da la posición en el alfabeto normal, tenemos que:

$$e(x) = (x + k) mod\ 26$$

donde k es el desplazamiento o corrimiento que elegimos. Por ejemplo, si sabemos que a la 'A' le corresponde la posición 0, a la 'A' cifrada le corresponde la posición k, a la 'B' cifrada la posición $k+1$, y así sucesivamente. Podemos pensar en que mapeamos el alfabeto como se ve en la Figura 1.

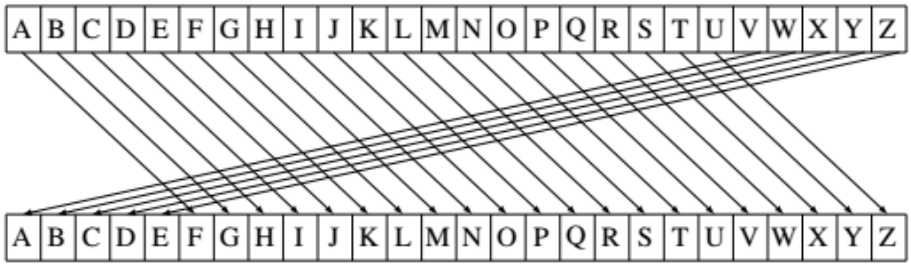

Figura 1: Cifrado de César.

Método

Para este cifrado, el encabezado de los métodos será:

```
/*Método que lee la cadena de entrada.*/
public String obtenCadena(Scanner sc);

/**
 * Trabaja con un atributo de la clase como cadena
 * origen y regresa la cadena cifrada.
 *
 * @param k El desplazamiento que se desea dar.
 * @return La cadena cifrada.
 */
 public String cifradoDeCesar(int k);
```

```java
/**
 * Trabaja con un atributo donde se encuentra la cadena
 * cifrada y regresa la cadena original.
 *
 * @param k El desplazamiento original.
 * @return La cadena descifrada sin blancos ni caracteres
 *         especiales.
 */
public String cifradoDeCesar(int k);
```

Los métodos tienen que ser públicos y respetar los encabezados, por lo que no pueden agregar **static**, ni parámetros, ni nada.

Cifrado de rieles

El cifrado de rieles es lo que se conoce como un cifrado de trasposición. En este tipo de cifrado se acomodan los caracteres del texto original en rieles, ignorando los blancos, caracteres especiales y dígitos, como se muestra en la Figura 2.

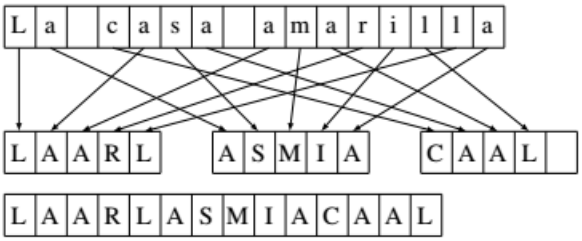

Figura 2: Cifrado de rieles.

Se escribe el texto a cifrar que se encuentra en la primera línea y se distribuye en cada uno de los rieles, eliminando espacios en blanco y caracteres especiales. Al final se pegan los rieles y eso corresponde al texto cifrado. Los caracteres alfabéticos se van acomodando en rieles, en este caso tres: el primer caracter en el primer riel, el segundo en el segundo, el tercero en el tercero, el cuarto en el primer riel y así sucesivamente, como se muestra en la segunda línea de la figura. La fórmula general para acomodarlos es:

$$texto_i \text{ va en el riel } i\,\%3$$

Al final se pegan los rieles, uno a continuación del otro, como se muestra en la tercera línea de la figura. Esto es lo que corresponde al texto cifrado.

Los encabezados de los métodos se dan a continuación:

```java
/**
 * Regresa los tres rieles pegados. Trabaja a partir
 * de un atributo donde se encuentra la cadena origen.
 *
 * @return Una cadena con los rieles pegados sin blancos.
 */
public String cifraRieles();

/**
 * Trabaja con un atributo donde se encuentra la cadena
 * y regresa la cadena original
 *
 * @return La cadena original, pero sin blancos ni caracteres
 *         especiales.
 */
public String descifraRieles();
```

Al igual que en el caso anterior, estos métodos tienen que ser públicos. No pueden agregar o quitar **nada** del encabezado de cada uno de ellos. Tienen que trabajar con atributos que no sean *static*.

Cifrado de Kama-sutra

Una de las primera descripciones de un cifrado por sustitución aparece en el Kama-sutra, un texto escrito en el siglo IV DC por Vatsayana, un estudioso de Brahmin, pero basado en manuscritos tan antiguos que se ubican en el siglo IV AC. El Kama-sutra recomienda que las mujeres deben estudiar 64 artes que incluyen cocinar, vestir, masaje y la preparación de perfumes. La lista también incluye algunas artes menos obvias como invocaciones, ajederez, empastado y carpintería. El arte que ocupa el lugar 45 en esa lista es *mlechita-vikalpa*, el arte de escribir secretamente, recomendado para ayudar a las mujeres a esconder sus amoríos. Una de las técnicas recomendadas involucra aparear las letras del alfabeto aleatoriamente, para después sustituir cada letra por su pareja. Por supuesto que el destinatario del mensaje tiene que conocer el apareamiento. A este tipo de cifrado se le conoce como de llave privada.

En la Figura 3 aparece un esquema de este tipo de cifrado. Para revolver el alfabeto se debe utilizar la clase java.util.Random (hay que hacer el ***import***). Al construir un nuevo objeto de esta clase, se crea un generador de números aleatorios. La sugerencia para este método se encuentra en el diagrama de Warnier a continuación.

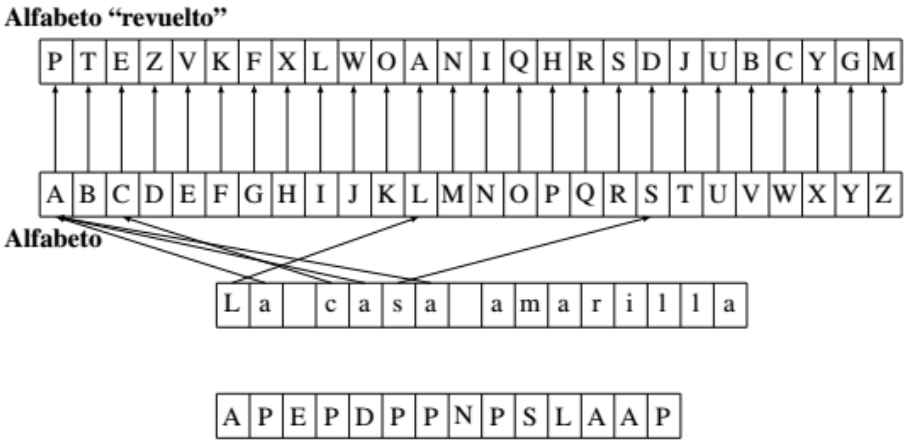

Figura 3: Cifrado de Kama-sutra.

- Revolver alfabeto
  - Principio
    - Copiar el alfabeto.
    - revuelto=cadena vacía.
    - Crear el generador de números aleatorios.
  - Acomodar letras (Mientras haya)
    - Elegir posición en el alfabeto.
    - Pegar esa letra a revueltos.
    - Eliminar esa letra del alfabeto.
  - Final
    - Entregar revueltos.

Notas

- Para crear un generador de números aleatorios hay dos opciones:

```
public Random(); /*Crea un generador verdaderamente
                   aleatorio. Cada vez que lo ejecuten
                   va a generar una nueva sucesión de
                   números aleatorios. */

public Random(long semilla); /*Usa la semilla que le den,
                               garantizando que siempre
                               da la misma secuencia.*/
```

  Es muy mala idea usar la primera opción, ya que, si algo está fallando, cada vez que ejecuten van a tener resultados distintos.

- Para elegir una posición pueden usar el método cuyo encabezado es:

```
public int nextInt(int mod);
```

  y obtiene un número aleatorio entre 0 y $mod - 1$.

Los encabezados de los métodos tienen que ser como sigue:

```
/**
 * Revuelve el alfabeto. Trabaja a partir del alfabeto, que
 * puede ser estático y constante.
 *
 * @return El alfabeto revuelto.
 */
public String revuelve();

/**
 * Trabaja con un atributo de la clase como cadena origen y
 * regresa la cadena cifrada.
 *
 * @param revuelto El alfabeto revuelto.
 * @return La cadena cifrada.
 */
public String cifradoKamaSutra(String revuelto);

/**
 * Trabaja con un atributo donde se encuentra la cadena
 * cifrada yregresa la cadena original.
 *
 * @param revuelto El alfabeto que se usó para cifrar.
 * @return La cadena descifrada, aunque sin blancos ni
 *         caracteres especiales
 */
public String descifraKamaSutra(String revuelto);
```