UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

Estructura de Datos y Algoritmos I

Actividad Asíncrona #04 Viernes

Cifrado César

(Pseudocódigo e Implementación)

Alumno:

Flores Ramírez Eduardo Amilcar

19 de marzo del 2021

Actividad Asíncrona #04 Viernes: Realizar su respectivo pseudocódigo y la implementación en un lenguaje distinto a C el cifrado César.

Pseudocodigo en Python:

```
#!/urs/bin/env phyton                    //ubicación del interprete de phyton
# -*- coding: utf -8 -*-                 //codificacion de caracteres empleados

    abc = 'abcdefghijklmnopqrstuvwxyz'   //se declara una variable, en donde se incluye el abecdario

def cifrar(cadena, clave):               //en esta parte se recibe la cadena y la clave
    text_cifrado = ''                    //se crea una variable en donde se va a almacenar el texto ya cifrado
    for letra in cadena:                 //se crea una iteración, en donde, letra, va a tomar los valores de cada letra en la cadena
        suma = abc.find(letra) + clave   //se realiza una suma, en donde se pide que nos devuelva la posición en donde se encuentra la letra, sumandole la clave numerica
        modulo = int(suma) % len(abc)    //en esta parte queremos conocer cual es el modulo, dado que se divide el modulo de la suma entre el total del abecedario
        text_cifrado = text_cifrado + str(abc[modulo])   //para finalmente codificar, en donde sumamos el texto cifrado mas la letra en el abecedario que tenga el valor de modulo
    return text_cifrado                  //aquí solo pedimos que regrese un resultado porque en esta parte las letras ya estan recorridas

def decifrar(cadena, clave):             //para poder descifrir el codigo se utiliza el mismo procedimiento solo que esta vez se resta el valor de la clave, y para
```

```
                                        implementarlo es exectamente lo mismo
                                        solo que en lugar de más, ponemos un
                                        menos y cambiamos el nombre por
                                        decifrar
    text_cifrado = ‘’
    for letra in cadena:
        suma = abc.find(letra) – clave      //aquí es donde cambia el más por un
                                            menos
        modulo = int(suma) % len(abc)
        text_cifrado = text_cifrado + str(abc[modulo])

    return text_cifrado

def main():
    c = str(raw_input(‘cadena a cifrar: ’)).lower()     //se pide la cadena, en
                                                        donde se cifra el mensaje,
                                                        como tambien se
                                                        corvierten en minusculas
                                                        las letras
    n = int(raw_input(‘clave numerica: ’))              //se pide el numero con el
                                                        que se va a cifrar
    print cifrar (c,n)                                  //se imprime el resultado
                                                        de la funcion cifrar en
                                                        donde se recibe la cadena
                                                        y el número
    cc = str(raw_input(‘cadena a decifrar: ’)).lower()  //en esta parte es lo
                                                        mismo de arriba solo que
                                                        aquí se cambian algunos
                                                        nombres y algunos
                                                        procedimientos pero la
                                                        finalidad es casi la misma
    cn = int(raw_input(‘clave numerica: ’))
    print decifrar(cc,cn)
```

if __name__ == ‘__main__’:

main()

Pseudocodigo en Python:

```
#!/urs/bin/env phyton
# -*- coding: utf -8 -*-

abc = ‘abcdefghijklmnopqrstuvwxyz’

def cifrar(cadena, clave):
    text_cifrado = ‘’
    for letra in cadena:
        suma = abc.find(letra) + clave
        modulo = int(suma) % len(abc)
        text_cifrado = text_cifrado + str(abc[modulo])

return text_cifrado

def decifrar(cadena, clave):
    text_cifrado = ‘’
    for letra in cadena:
        suma = abc.find(letra) - clave
        modulo = int(suma) % len(abc)
        text_cifrado = text_cifrado + str(abc[modulo])

    return text_cifrado

def main():
    c = str(raw_input(‘cadena a cifrar: ’)).lower()
    n = int(raw_input(‘clave numerica: ’))
    print cifrar (c,n)
    cc = str(raw_input(‘cadena a decifrar: ’)).lower()
    cn = int(raw_input(‘clave numerica: ’))
    print decifrar(cc,cn)

if __name__ == ‘__main__’:
    main()
```

Resultado final (Consola):

El programa primero pide el código, después la clave numérica y al fina lo cifra. Un caso muy parecido al descifrar primero pide el mensaje cifrado después la clave numérica y finalmente lo descifra.