Apellido y Nombres	Legajo	Calificación

Condiciones:
Bla Bla Bla**primerParcial**.

1. Implemente una función que contenga el algoritmo de cifrado **ROT13**.
   El algoritmo consiste en sustituir cada letra por una que se encuentra trece posiciones por delante. Por ejemplo la A se reemplaza por la N, la B por la O y así sucesivamente. Para las últimas trece letras la secuencia se invierte. A continuación se muestra la tabla de las equivalencias entre las letras.

A	B	C	D	E	F	G	H	I	J	K	L	M
N	O	P	Q	R	S	T	U	V	W	X	Y	Z

   Ejemplo:

Texto sin cifrar	H	O	L	A		M	U	N	D	O
Texto cifrado	U	B	Y	N		Z	H	A	Q	B

   Prototipo de la función:

   int rot_13 (char *dataPtr)

   Donde **dataPtr** es el puntero al mensaje a cifrar (string) y donde se colocará el mensaje cifrado y devuelve:

   - **-1** Si el mensaje contiene un caracter **distinto** de una **letra mayúscula** o un **espacio**.
   - Un **número positivo** indicando la cantidad de caracteres convertidos sin tener en cuenta los espacios ni el **\0**

2. Implemente una función que realice la validación de una CBU (Clave Bancaria Uniforme).
   La CBU está formado de la siguiente manera.

0	1	4	0	1	2	5	6	5	5	6	5	4	1	8	5	4	7	6	5	4	3
E_0	E_1	E_2	S_0	S_1	S_2	S_3	D_0	C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}	C_{11}	C_{12}	D_1

   Donde:

   - **E0 a E2**: Es el número de la entidad bancaria.
   - **S0 a S3**: Es el número de sucursal.
   - **D0**: Es el dígito verificador de E y S
   - **C0 a C12**: Es el número de cuenta.
   - **D0**: Es el dígito verificador de C

Los dígitos verificadores de la clave bancaria única se calculan de la siguiente forma:

$R_0 = E_0*9 + E_1*7 + E_2*1 + S_0*9 + S_1*7 + S_1*1 + S_2*3$
$D_0 = 10 - (R_0 \% 10)$

$R_1 = C_0*9 + C_1*7 + C_2*1 + C_3*3 + C_4*9 + C_5*7 + C_6*1 + C_7*3 + C_8*9 + C_9*7 + C_{10}*1 + C_{11}*3 + C_{12}* 9$
$D_1 = 10 - (R_1 \% 10)$

La función tiene el siguiente prototipo:

int cbu_validar (char *dataPtr)

Donde **dataPtr** es el puntero la clave bancaria uniforme a validar terminada en '**\0**'

Devuelve:

- **-1** Si la CBU pasada no tiene 22 caracteres.
- **-2** Si alguno de los caracteres de la CBU no es un número.
- **-3** Si el dígito verificador D0 no corresponde.
- **-4** Si el dígito verificador D1 no corresponde.

3. Implemente una función que obtenga el dígito verificador de un número de CUIT pasado como parámetro, el cálculo se realiza utilizando el algoritmo **módulo11**.
El prototipo de la función es el siguiente:

int cuit_validar (char *cuit);

El parámetro **cuit** es un puntero al vector que contiene el número de CUIT terminado en '**\0**'

Devuelve:

- Un **número positivo** indicando el dígito verificador.
- **-1**: cuando la cantidad de dígitos es distinto de 10
- **-2**: Indica que el número de CUIT es inválido (contiene algo distinto a números)

Algoritmo módulo 11:

- Multiplique los dígitos Desde el menos significativo por la serie 2,3,4,5,6,7.
- Sume el resultado de las multiplicaciones anteriores.
- Calcule el módulo 11 de la suma anterior.
- Al resultado anterior restele 11
  - si el resultado es **menor que 10** lo obtenido es el dígito verificador
  - si el resultado es **10** el dígito verificador es **9**.
  - Si el resultado es **11** el dígito verificador es **0**.

Ejemplo:

CUIT	2	0	1	2	3	4	5	6	7	8	suma	%11	dígito
Valor a multiplicar por digito	X_5	X_4	X_3	X_2	X_7	X_6	X_5	X_4	X_3	X_2		148%11	11-5
Resultado de la multiplicación	10	0	3	4	21	24	25	24	21	16	=148	=5	6

4. Explique qué entiende por variable, nombre los tipos que conozca y sus características. De ejemplos de uso.