NOM		DATA	**11 / 4 / 2019**	QUALIFICACIÓ
ÀREA/MATÈRIA	**DAW - M03 - UF2**	CURS	**2018 - 2019**	

Feu captures de pantalla per a documentar l'examen. A l'acabar-ho, trameteu-ho a carles.olive@fje.edu i jordibinefa@fje.edu, adjuntant un arxiu en **pdf** demostrant els punts claus de les respostes i un arxiu **.zip** (o **.tar.gz**, **No** s'accepten **.7z** o **.rar**) amb els codis i **sense** els arxius **executables**. Aquest examen té més de 10 punts, escolliu les preguntes que considereu més adients.

RA1-1)***(2 punts)*** Depuració
RA1-1-1)***(0,3 punts)*** Compileu **cc03.c**, adjunt a aquest examen, per a poder ser depurat pel **gdb**. El nom de l'executable ha de ser **ex01**.
RA1-1-2)***(0,3 punts)*** Emprant el **gdb**, visualitzeu el codi des de la línia **20** fins a la **34**.
RA1-1-3)***(0,4 punts)*** Poseu un punt de ruptura per conèixer el valor de la variable **nSuma** a cada iteració.
RA1-1-4)***(1 punt)*** Visualitzeu el valor de **nSuma** a les **tres** primeres iteracions.

RA1-2)***(1,5 punts)*** Dividiu **cc03.c** en cinc arxius: **cc03main.c** (a on hi va la funció principal), **cc03calcul.c** (implementació de les funcions de càlcul), **cc03calcul.h** (declaració de prototipus de les funcions de càlcul), **cc03presentacio.c** (implementació de les funcions de presentació), **cc03presentacio.h** (declaració de prototipus de les funcions de presentació **i els #define**). **Documenteu** com feu la compilació.

RA1-3)***(2 punts)*** Desenvolupeu la funció recursiva **nSumaElementsDelVector(int* nV,int nQuants)** que retorna la suma de tots els elements del vector. Aquest és l'algorisme proposat:

```
Input: an array A of n numbers
Output: the sum of the numbers in an array

Algorithm Sum(A, n)
    if n=1
        return A[0]
    s = Sum(A, n-1)   /* recurse on all but last */
    s = s + A[n-1]    /* add last element */
    return s
```

Aprofiteu el codi **ex03previ.c** i deseu-lo com a **ex03.c**. Exemple d'execució:

```
$ ./ex03
El vector { 1 } suma 1
El vector { 1, 20 } suma 21
El vector { 1, 20, 3 } suma 24
El vector { 1, 20, 3, 40 } suma 64
El vector { 1, 20, 3, 40, 5 } suma 69
```

RA1-4-1)***(2 punts)*** Desenvolupeu la funció **int nComptaNumerosSenars(int* nV,int nQuants)** que compta el nombre de números senars del vector passat per referència. Aprofiteu el codi **ex04previ.c** i deseu-lo com a **ex04.c**. Exemple d'execució:

```
$ ./ex04
El vector { 1 } té 1 números senars
El vector { 1, 20 } té 1 números senars
El vector { 1, 20, 3 } té 2 números senars
El vector { 1, 20, 3, 40 } té 2 números senars
El vector { 1, 20, 3, 40, 5 } té 3 números senars
```

RA1-4-2)*(2 punts)* Desenvolupeu la funció **nComptaNumerosSenars(nV,nQ)** que compta el nombre de números senars de la llista passada . Aprofiteu el codi **ex04previ.py** i deseu-lo com a **ex04.py**. Exemple d'execució:

```
$ ./ex04.py
La llista [ 1 ] té 1 números senars
La llista [ 1 20 ] té 1 números senars
La llista [ 1 20 3 ] té 2 números senars
La llista [ 1 20 3 40 ] té 2 números senars
La llista [ 1 20 3 40 5 ] té 3 números senars
```

RA1-5)*(2 punts)* Deseu **ex05BibliotecaPrevia.*** com a **ex05Biblioteca.***. Modifiqueu **ex05Biblioteca.c** per a que permeti desencriptar. Aprofiteu el codis a **ex05Previ.zip**. Aquesta és la lògica de funcionament:

```
$ ./e
Escriviu una frase sense accents, ni ç, l·l o ñ, per a encriptar-la: La Jugada Mestra de qui ho faci
Frase encriptada: XdQLojdgdQwtfuydQgtQeopQkaQhdbp
Frase desencriptada: La Jugada Mestra de qui ho faci
```

```c
#include <stdio.h>
#include "ex05Biblioteca.h"

int main(){
    char szCadena[N],szCadenaEncriptada[N],szCadenaDesencriptada[N];

    vPreguntaFrase(szCadena);
    vCodifica(szCadena,szCadenaEncriptada,'e');
    printf ("Frase encriptada: %s \n",szCadenaEncriptada);
    vCodifica(szCadenaEncriptada,szCadenaDesencriptada,'d');
    printf ("Frase desencriptada: %s \n",szCadenaDesencriptada);

    return 0;
}
```

RA1-6)*(1,5 punts)* Desenvolupeu la funció recursiva **nEnesimNumeroNaturalSenar(nQuin)** que troba l'enèsim número natural senar.

Aprofiteu el codi **ex06previ.py** i deseu-lo com a **ex06.py**. Aquesta és la lògica de funcionament:

Si **nQuin** = 1 → **nEnesimNumeroNaturalSenar(nQuin)** = 1
Si **nQuin** != 1 → **nEnesimNumeroNaturalSenar(nQuin)** = 2+**nEnesimNumeroNaturalSenar(nQuin-1)**

```
$ ./ex06.py
Enèssim número natural senar (> 0): 4
Element 4 de la llista de números naturals senars: 7
Element 4 de la llista de números naturals senars (sense recursivitat): 7
```

Què fa el programa?: Números naturals senars: 1, 3, 5, **7**, 9, 11, ... (el quart és el 7)

Molta sort a totes i tots !!!!