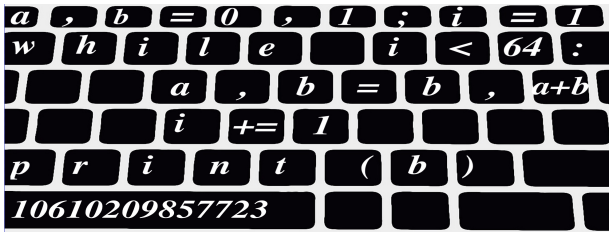

FUNDAMENTOS DE COMPUTACIÓN
Grado en ingeniería en tecnologías Industriales
Universidad de Cantabria

Examen Extraordinario (parte escrita)
15 de Febrero del 2021

Nombre y apellidos:

1. [10 %] Douglas R. Hofstadter, ganador del premio Pulitzer de 1980 con la obra *Gödel, Escher, Bach: an Eternal Golden Braid*, definió las siguientes funciones mutuamente recursivas:

hembra.py

```
def hembra(n):
    if n == 0:
        return n
    f = hembra(n-1)
    return n+1 - macho(f)
```

macho.py

```
def macho(x):
    if x == 0:
        return 0
    m = macho(x-1)
    return x-1 - hembra(m)
```

¿Cuál es la salida de `hembra(3)` `macho(4)` y `hembra(-1)`?

`hembra(3)`	`macho(4)`	`hembra(-1)`

2. Considera este fragmento de código

```
cadena = 'Esparadrapo malo'
i = 1
seguimos = 'Si'
while seguimos:
    letra = cadena[i-1]
    print(letra)
    if letra == 'a' or i - 1 == len(cadena):
        seguimos = {}
    i += 1
```

¿Qué valor tienen, al final de la ejecución, las variables `i`, `letra` y `seguimos`?

3. [10 %] ¿Qué resultados se obtendrán al evaluar las siguientes expresiones? Calcula primero a mano el valor resultante de cada expresión y comprueba, con la ayuda del ordenador, si tu resultado es correcto.

```
>>> True == True != 1
SOLUCION:
>>> False >= 0
SOLUCION:
>>> type(True + 2)
SOLUCION:
>>> 1 or abcd
SOLUCION:
>>> 0 or abcd
SOLUCION:
>>> 1 is in [3,2,1]
SOLUCION:
```

```
>>> 0 and abcd
SOLUCION:
>>> 1 and abcd
SOLUCION:
>>> print(2**0 * False + 2**2**True * False+2**True,"Torre", False)
SOLUCION:
>>> a = 40;  a%20 == 0 or (not a %5== 0 and a%4==0)
SOLUCION:
>>> a = 40;  (a%20 == 0 or not a %5== 0) and a%4==0
SOLUCION:
>>> 1 is not [3,2,1.]
SOLUCION:
```

4. [10 %] Dadas dos listas `L1` y `L2`, queremos eliminar los elementos de L1 que están en L2:

```
>>> quitar_comunes([1,2,3,4], [1,2,5,6])
[3, 4]
quitar_comunes([2,5,-1,5], [7,-1,2])
[5]
```

Los dos programas siguientes tratan de implementarlo.

```
1 def quitar_comunes1(L1, L2):
2     for x in L1:
3         if x in L2:
4             L1.remove(x)
```

```
1 def quitar_comunes2(L1, L2):
2      L3 = L1
3     for x in L3:
4         if x in L2:
5             L1.remove(x)
```

Explica brevemente los dos programas. ¿Son correctos? ¿En qué se diferencian ?

5. [10 %] Implementa la función `encontrar_maximo` de dos formas distintas: iterativamente, usando el bucle `while` y recursivamente, usando específicamente la estrategia 'divide-y-venceras'. La entrada es una lista de números(float) de longitud al menos 1, y la salida el máximo de los elementos

   1. Iterativamente [3 %]

```
def encontrar_maximo(mi_lista):
 '''   No se puede modificar mi_lista, y debes usar un bucle <while> '''
```

   2. Recursivamente [7 %]

```
def encontrar_maximo(mi_lista):
 '''  No se puede modificar mi_lista, y debes usar  recursividad, dividiendo  la lista de
     entrada en dos mitades de igual longitud (o más 1), encontrar el máximo de ambas
     mitades y luego devolver el máximo de las dos.'''
```

6. [11 %] Implementa la función `redactar_correo` de acuerdo con su especificación. No puedes usar el método `replace`.

```
def redactar_correo(lista_correo, caracter):
   ''' Entrada: - lista_correo es una lista. Cada elemento es una lista de cadenas.
                - caracter: una cadena de longitud mayor que uno.
       Salida: muta la lista_correo como se detalla: cada cadena conteniendo el carácter es
       reemplazado por x's.  La cadena de las x's tiene la misma longitud que la longitud de
       la cadena original. lista_correo  pero no se devuelve. '''
```

```
lista_correo: [ ['llamar', 'en', 'enseguida'], ['encuentro', '3', 'por favor', 'tarde' ], [] ]

(1)caracter : 'en'
redactado: [['llamar', 'xx', 'xxxxxxxxx'], ['xxxxxxxxx', '3', 'por favor', 'tarde'], [] ]
(2)caracter : 'e'
redactado: [['llamar', 'xx', 'xxxxxxxxx'], ['xxxxxxxxx', '3', 'por favor', 'xxxxx'], [] ]
(3) caracter : 'tu'
redactado:  [ ['llamar', 'en', 'enseguida'], ['encuentro', '3', 'por favor', 'tarde' ], [] ]
```

7. [9 %] A continuación aparecen tres funciones f, g y h de Python que toman como argumento una lista L de números de longitud n.

```
def f(L):
    suma = 0
    i=1
    while i < len(L):
         suma = suma + L[i]
         i = i * 2
    return suma
```

```
def g(L):
    i = 2**(len(L))
    suma = 100
    while i >= 0:
         suma += len(L)
         i -= -1
    return suma
```

```
def h(L):
    suma = 0
    for x in L:
        for i in range(1, 101):
            if x >= i:
                suma = suma + 1
    return suma
```

¿Cuál de las siguientes afirmaciones describe con mayor precisión cómo crece el tiempo de ejecución de cada una de ellas a medida que n crece? No contestar al azar, se penaliza con 3 % puntos cada respuesta incorrecta.

(a) Crece linealmente, como n
(b) Crece cuadráticamente como n^2
(c) Crece menos que linealmente
(d) Crece más que cuadráticamente

SOLUCIÓN para f:	SOLUCIÓN para g:	SOLUCIÓN para h:

8. [10 %] Escribe un programa para resolver una clásica adivinanza: 'En una granja hay A cabezas y B patas entre gallinas y conejos, ¿cuántas gallinas y cuántos conejos hay en la granja?'

```
A = int(input('Dame numero de cabezas: '))
B= int(input('Dame numero de patas: '))
```

9. [5 %] Transformar a binario (usando a lo sumo 16 bits) el número decimal 39.31.

10. [5 %] Los *bytes* que figuran a continuación constituyen la codificación UTF-8 de un texto. Se pide:

    1. determinar cuántos caracteres contiene,
    2. escribir en notación hexadecimal los *code points* de esos caracteres.

1	1	1	0	0	0	1	0
1	0	1	1	1	1	1	1
1	0	1	0	0	0	1	1
1	1	0	0	0	0	1	0
1	0	0	0	0	1	0	0
1	0	0	1	0	1	0	1
1	0	1	1	1	1	0	1
1	1	1	1	0	0	1	0
1	0	0	0	0	1	0	0
1	0	1	1	0	1	0	1
1	0	1	0	0	0	1	0
1	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0
0	0	1	1	1	1	1	1

11. [10 %] Supongamos que disponemos de la clase `class Punto` con atributos números x e y para las coordenadas, y los siguientes métodos:

```
def __init__(self,x,y):
  """ Crear un Punto."""
def Dist(self, otro):
  """ Devuelve la distancia  del punto self al punto  otro."""
def VecinoAleatorio(self):
   """ Devuelve un Punto aleatorio con distancia <= 2 al Punto self """
```

El siguiente programa simula un robot que comienza en $(0, 0)$ y salta aleatoriamente de un punto a otro en el plano.

```
P = Punto(0,0)
Z = P
t = 0
while P.Dist(Z) < = 100 and t < 100000:
    P = P.VecinoAleatorio()
    t += 1
```

1. ¿Es posible que justo después de que termine este código t sea inferior a 100000? Explica tu respuesta en no más de tres lineas:

2. Supongamos que cuando el robot realiza un salto desde el Punto P a un Punto con distancia mayor que 1, *tu hermano pequeño* tira del robot para que regrese al punto P. Escribir un código que simule este proceso para el robot comenzando en (0,0) e intentando 100 saltos.