Fundamentos de la Programación - Grado en Física - Curso 2020/2021

Examen final C2 (25/02/2021)

Instrucciones

- Tienes **2 horas y 30 minutos** para realizar el examen. Lee tranquilamente el examen y decide por dónde empezar.
- No puedes comunicarte con nadie (excepto con el profesor) durante el examen.
- **Importante:** Pon tu DNI, nombre y apellidos al principio de cada ejercicio que entregues.
- Debes comprimir tu examen con todos los ejercicios que hayas realizado en un archivo `zip` cuyo nombre debe ser tu DNI (`NUMERO_DNI.zip`).
- Debes entregar dicho archivo comprimido a través de la tarea creada en MoodleUA, donde has encontrado este enunciado. Dicha aplicación se cerrará automáticamente una vez transcurrido el tiempo del examen, de modo que debes tener cuidado en no retrasarte.

Ejercicios

1. (ej1.py) **(3 puntos)** En un archivo de texto llamado `notas.dat` guardamos las calificaciones obtenidas por m estudiantes (a cada estudiante le corresponde una línea del archivo) en n ejercicios entregados semanalmente (cuando un ejercicio no se ha entregado la calificación es -1). Las notas son números enteros.

   a) **(1 punto)** Diseña una función `leeNotas()` que lea el archivo y devuelva las notas de los estudiantes en una matriz de dimensión $m \times n$. Si el archivo no está disponible esta función debe devolver una lista vacía.

   b) **(1 punto)** Se considera que un alumno abandonó en la semana x si no ha entregado ningún ejercicio desde entonces (si un alumno no ha entregado nunca ningún ejercicio, abandonó en la semana 0). Diseña una función `abandono(notasAlu)` a la que se le pasa como parámetro una lista con las notas de un alumno y devuelve la semana en la que abandonó o `None` si no abandonó.

   c) **(1 punto)** El programa principal debe imprimir por pantalla el número de abandonos en función de la semana. Las semanas en las que no hubo abandonos no deben imprimirse.

   Ejemplo: Si el archivo `notas.dat` (se adjunta con este enunciado) contiene:

```
 5  6 -1  3 -1 -1 -1 -1  0 -1
-1 -1 -1 -1 -1 -1 -1 -1 -1 -1
 9  9  9  9  9  9  9  9  9  9
 1  2 -1 -1  3  7  8 -1 -1 -1
 3  6  7  8  8  7  6  9  2 -1
-1  9 -1  9 -1  9 -1  9 -1  9
 8  2 -1 -1  3  7  8 -1 -1 -1
 7  2 -1 -1  3  7  8 -1 -1 -1
```

   la salida del programa debe ser

```
0: 1
7: 3
9: 2
```

2. (ej2.py) **(3 puntos)** El número de formas diferentes de dividir un conjunto de n elementos en k subconjuntos (particionar el conjunto) se denota con $\left\{ \begin{array}{c} n \\ k \end{array} \right\}$ y satisface la fórmula recursiva

$$\left\{ \begin{array}{c} n \\ k \end{array} \right\} = \left\{ \begin{array}{c} n-1 \\ k-1 \end{array} \right\} + k \left\{ \begin{array}{c} n-1 \\ k \end{array} \right\}$$

   El valor de $\left\{ \begin{array}{c} n \\ 1 \end{array} \right\}$, al igual que el de $\left\{ \begin{array}{c} n \\ n \end{array} \right\}$, es 1.

   a) **(1.5 puntos)** Diseña una función, `particiones(n,k)`, que devuelva el valor de $\left\{ \begin{array}{c} n \\ k \end{array} \right\}$.

b) **(1.5 puntos)** Al programa principal se le debe pasar por línea de comandos un número n y debe escribir en un fichero de texto llamado `particiones_n.txt` (`n` es el número pasado al programa) el número de particiones posibles de un conjunto de n elementos en el formato del ejemplo.

Ejemplo: Si ejecutamos el programa con la orden

```
$ python3 ej2.py 6
```

debe haberse creado el fichero `particiones_6.txt` que contendrá

```
1 1
2 31
3 90
4 65
5 15
6 1
```

3. (ej3.py) **(2 puntos)** Utilizando las librerías *numpy*, *matplotlib.pyplot* y el método *odeint* de la librería *scipy.integrate*, resuelve numéricamente la ecuación diferencial

$$\frac{dy}{dx} = \frac{y}{x} - y^2 \quad 1 \leq x \leq 2$$

con la condición inicial $y(1) = 1$.

Guarda la solución en una gráfica dibujada con 100 puntos en un archivo que se llame `solucion.png`.

4. (ej4.py) **(2 puntos)** Nos han pasado el siguiente programa en Fortran 90 que es capaz de encontrar el número de Hardy-Ramanujan[1]. Escribe un programa en Python que realice este mismo cálculo.

```
program hardy_ramanujan
    implicit none
    integer :: hr,a,b,c,d

    hr = 0
    do a = 1, 100
        do b = a+1, 100
           do c = a+1, 100
               if (c /= b) then
                   do d = c+1, 100
                       if (a**3 + b**3 == c**3 + d**3) then
                           hr = a**3 + b**3
                           exit
                       end if
                   end do
               end if
               if (hr /= 0) then
                   exit
               end if
           end do
           if (hr /= 0) then
               exit
           end if
        end do
        if (hr /= 0) then
            exit
        end if
    end do
    print*, hr, a, b, c, d
end program hardy_ramanujan
```

[1] El número de Hardy-Ramanujan es el menor número entero que puede ser expresado como la suma de dos cubos perfectos de dos maneras distintas ($a^3 + b^3 = c^3 + d^3$).