Examen Final

Fundamentos de la Informática

13-Diciembre-2019

Tiempo: 2h15

Rellena la siguiente información:

Nombre completo: Jorge González Cardelús

Grupo: 1º - GITT

NOTA. Al terminar, el estudiante debe subir el notebook en formato PDF y en formato IPYNB al Moodle en este orden (a la sección con nombre "EXAMEN 13/12/2019").
Para convertir el notebook a formato PDF pulsar Ctrl+P y seleccionar "Salvar a PDF" en el desplegable de selección de impresoras

Ej1. Lectura de ficheros (2 ptos.)

Junto al enunciado se han distribuido dos ficheros "Meses.txt" y "Descuentos.txt". "Meses.txt" contiene los diferentes meses del año y un identificador de descuento para cada uno. "Descuentos.txt" contiene el importe exacto de cada descuento.

Se pide:

1. Leer el fichero "Meses.txt" y generar un diccionario llamado "dic_month" en el que las claves sean los nombres de los meses y los valores el identificador del descuento:

```
{'Enero': 'd1',
 'Febrero': 'd3',....}
```

2. Leer el fichero "Descuentos.txt" y generar un diccionario llamado "dic_discount" en el que las claves sean los identificadores de descuento y el valor sea el valor numérico de dicho descuento:

```
{'d0': 0.0,
 'd1': 10.0,....}
```

Solución:

In [1]:

```
dic_month = {}

with open('Meses.txt', 'r') as f:
    lines = f.readlines()
    for i in range(1, len(lines)):
        parts = lines[i].split(';')
        dic_month[parts[1]] = parts[2].rstrip()

dic_discount = {}

with open('Descuentos.txt', 'r') as f:
    lines = f.readlines()
    for i in range(1, len(lines)):
        parts = lines[i].split('^')
        dic_discount[parts[0]] = float(parts[2].rstrip())

print(dic_month)
print(dic_discount)
```

```
{'Enero': 'd1', 'Febrero': 'd3', 'Marzo': 'd0', 'Abril': 'd2', 'Mayo': 'd1',
'Junio': 'd3', 'Julio': 'd0', 'Agosto': 'd2', 'Septiembre': 'd3', 'Octubre':
'd4', 'Noviembre': 'd5', 'Diciembre': 'd2'}
{'d0': 0.0, 'd1': 10.0, 'd2': 20.0, 'd3': 25.0, 'd4': 15.0, 'd5': 12.0}
```

Ej2. Combinar diccionarios (2 ptos.)

Utilizar los diccionarios generados en el apartado anterior (si no se ha conseguido, definirlos a mano) para construir un nuevo diccionario llamado "dic_final" en el que las claves sean el nombre de los meses del año y el valor sea una tupla en el que el primer elemento sea el identificador del descuento y el segundo elemento sea el importe exacto de cada descuento:

```
{'Enero': ("d1", 10.0),
 'Febrero': ("d3", 25.0) ,....}
```

Solución:

In [2]:

```
dic_final = {}

for month in dic_month.keys():
    discount_id = dic_month[month]
    discount_val = dic_discount[discount_id]

    dic_final[month] = (discount_id, discount_val)

print(dic_final)
```

```
{'Enero': ('d1', 10.0), 'Febrero': ('d3', 25.0), 'Marzo': ('d0', 0.0), 'Abri
l': ('d2', 20.0), 'Mayo': ('d1', 10.0), 'Junio': ('d3', 25.0), 'Julio': ('d
0', 0.0), 'Agosto': ('d2', 20.0), 'Septiembre': ('d3', 25.0), 'Octubre': ('d
4', 15.0), 'Noviembre': ('d5', 12.0), 'Diciembre': ('d2', 20.0)}
```

Ej3. Salvar a fichero (1 pto.)

Utilizar el diccionario generado en el apartado anterior (si no se ha conseguido, definirlo a mano), para salvar su información en un fichero de tipo CSV llamado "Mes-Descuentos.csv", cuya primera línea debe incluir una cabecera con los nombres "mes","id_descuento","valor". Y el separador debe ser el caracter "|":

```
mes|id_descuento|valor
Enero|d1|10.0
Febrero|d3|25.0
```

Solución:

In [3]:

```
with open('Mes-Descuentos.csv', 'w+') as f:
    header = 'mes|id_descuento|valor\n'
    f.writelines(header)

    for month in dic_final.keys():
        month_parts = dic_final[month]
        discount_id = month_parts[0]
        discount_val = month_parts[1]

        month_line = month + '|' + discount_id + '|' + str(discount_val) + '\n'
        f.writelines(month_line)

# Comprobación
import pandas as pd
df = pd.read_csv('Mes-Descuentos.csv', sep='|')
df.head(12)
```

Out[3]:

	mes	id_descuento	valor
0	Enero	d1	10.0
1	Febrero	d3	25.0
2	Marzo	d0	0.0
3	Abril	d2	20.0
4	Mayo	d1	10.0
5	Junio	d3	25.0
6	Julio	d0	0.0
7	Agosto	d2	20.0
8	Septiembre	d3	25.0
9	Octubre	d4	15.0
10	Noviembre	d5	12.0
11	Diciembre	d2	20.0

Ej4. Genera cualquier polinomio (2 ptos.)

Preguntar al usuario por el grado del polinomio que quiere pintar. A continuacion, el código debe preguntar al usuario por tantos coeficientes como grado tenga el polinomio. El código debe funcionar para cualquier grado (no es necesario chequear si el usuario introduce grados negativos, letras o símbolos). Al final, el código debe imprimir la función solicitada por el usuario:

```
Introduzca el grado del polinomio: 3
  - Introduzca el coeficiente grado 0:0
  - Introduzca el coeficiente grado 1:1
  - Introduzca el coeficiente grado 2:2
  - Introduzca el coeficiente grado 3:3
El polinomio solicitado es:
f(x)=3x^3+2x^2+1x^1+0
```

Solución:

In [4]:

```
def num_request(request):
    num = 0
    while True:
        try:
            num = int(input(request))
            break
        except ValueError:
            print("Por favor introduzca un número")

    return num

grado = num_request("Por favor, introduzca el grado del polinomio: ")
coefs = []
for i in range(grado):
    coef = num_request('  - Introduza el coef. del grado %s: ' % (str(i)))
    coefs.append(coef)

def print_polynomio(coefs):
    print('f(x) = ', end='')
    for i in reversed(range(len(coefs))):
        coef = coefs[i]

        if coef < 0:
            print('%s' % (str(coef)), end='')
        else:
            print('+%s' % (str(coef)), end='')

        if i > 0:
            print('x^%s ' % (str(i)), end='')
    print()


print()
print("El polinomio solicitado es")
print_polynomio(coefs)
```

```
Por favor, introduzca el grado del polinomio: 5
  - Introduza el coef. del grado 0: -1
  - Introduza el coef. del grado 1: 2
  - Introduza el coef. del grado 2: -3
  - Introduza el coef. del grado 3: 4
  - Introduza el coef. del grado 4: -5

El polinomio solicitado es
f(x) = -5x^4 +4x^3 -3x^2 +2x^1 -1
```

Ej5. Calcula el polinomio (2 ptos.)

Construye una función que responda a esta difinición:

```
def apply_polynomio(ejex,coeficientes):
    ...
    ...
return ejey
```

Dónde la variable "ejex" reciba una lista de valores para la variable X y la variable "coeficientes" reciba la lista de coeficientes seleccionados por el usuario (no es necesario haber realizado el apartado anterior para hacer este ejercicio). La variable devuelta "ejey" debe ser una lista de valores resultado de aplicar el polinomio definido por los coeficientes a cada uno de los valores de la lista "ejex".

```
ejex=[0,1,2,3]
coeficientes=[1,2,3,4]
ejey=apply_polynomio(ejex,coeficientes)
print(ejey)
[1, 10, 49, 142]
```

Solución:

In [5]:

```
def calc_image(x, coefs):
    y = 0
    for i in reversed(range(len(coefs))):
        coef = coefs[i]
        y += coef * (x ** i)

    return y

def apply_polynomio(ejex, coefs):
    ejey = []
    for x in ejex:
        ejey.append(calc_image(x, coefs))

    return ejey

ejex = [0,1,2,3]
print(ejex)
coefs = [1,2,3,4]
print_polynomio(coefs)
ejey = apply_polynomio(ejex, coefs)
print(ejey)
```

```
[0, 1, 2, 3]
f(x) = +4x^3 +3x^2 +2x^1 +1
[1, 10, 49, 142]
```

Ej6. Visualiza el polinomio (1 pto.)

Construye una función con la siguiente definición:

```
def plot_polynomio(ejex,ejey,kind):
```

Que visualice la función descrita por las listas de valores "ejex" y "ejey". El tipo de gráfico utilizado dependerá del parámetro "kind" que puede tomar los siguientes valores:

```
kind="line"
kind="scatter"
kind="bar"
```

Cuando la variable "kind" tome el valor "line" imprimirá el polinomio con una línea contínua negra utilizando cruces para las marcas.
Cuando la variable "kind" tome el valor "scatter" imprimirá el polinomio como una sucesión de puntos con forma circular de color rojo.
Cuando la variable "kind" tome el valor "bar" imprimirá el polinomio como un diagrama de barras verticales de color magenta.
Cuando la variable "kind" no reciba ningún valor por parte del usuario, por defecto, tomará el valor "line".
Cuando la variable "kind" reciba un valor no reconocible, por defecto, tomará el valor "line"

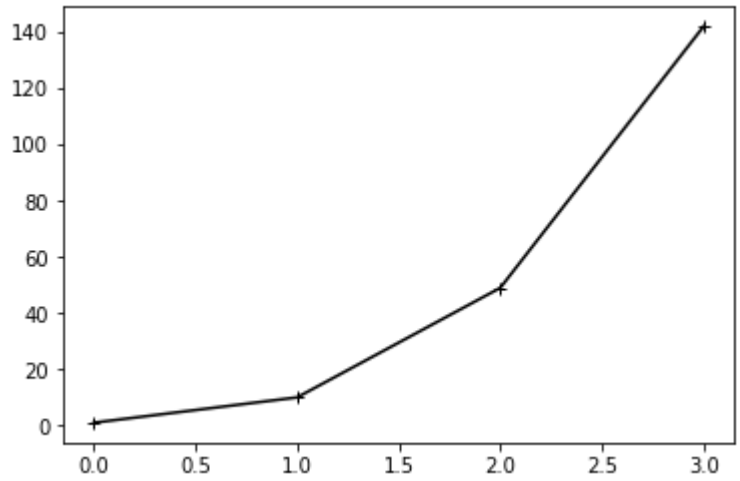
In [6]:

```
import matplotlib.pyplot as plt

def plot_polynomio(ejex, ejey, kind="line"):
    allowed_kinds = ["line", "scatter", "bar"]

    if kind not in allowed_kinds:
        kind = "line"

    if kind == "line":
        plt.plot(ejex, ejey, color='black', linestyle='-', marker='+')
    elif kind == "scatter":
        plt.scatter(ejex, ejey, color='red', marker='o')
    elif kind == "bar":
        plt.bar(ejex, ejey, color='magenta')

    plt.show()

plot_polynomio([0,1,2,3],[1, 10, 49, 142])
```

```
<Figure size 640x480 with 1 Axes>
```

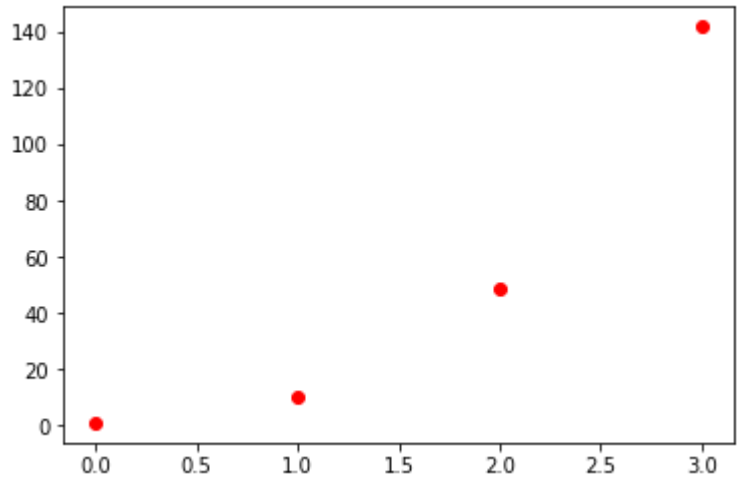
In [7]:

```
plot_polynomio([0,1,2,3],[1, 10, 49, 142],kind="cualquiercosa")
```



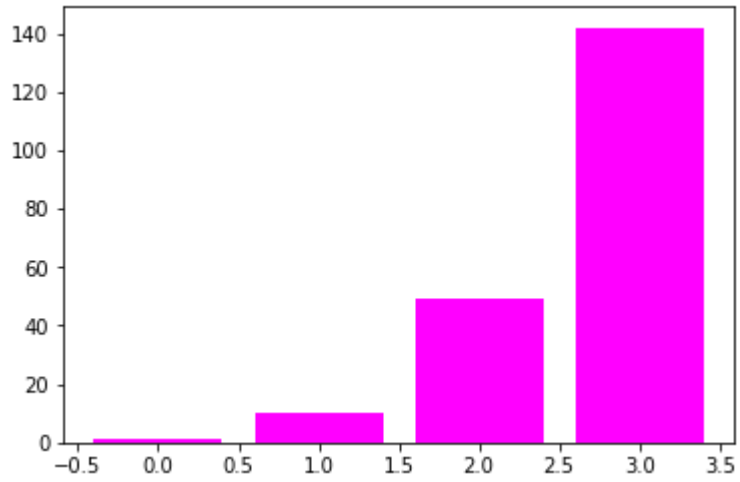
In [8]:

```
plot_polynomio([0,1,2,3],[1, 10, 49, 142],"scatter")
```



In [9]:

```
plot_polynomio([0,1,2,3],[1, 10, 49, 142],"bar")
```



Solución:

In [ ]:

```

```

In [ ]:

```

```