Bootcamp

April 1, 2022

**1 Clase 31-03-2022**

**1.0.1 Ejercicio 1**

Escriba un programa que administre una cuenta bancaria, usando una bitácora de operaciones. La bitácora de operaciones tiene la siguiente forma: D 100 R 50

• D 100 significa que depositó 100 pesos

• R 50 significa que retiró 50 pesos

[ ]: cuenta=0

**while True**:

x=input("Ingrese la opción que quiere ejecutar: ")

x=x.split()

**if** (x[0]=="D") **or** (x[0]=="d" ):

cuenta+=int(x[1])

print ("Deposito", int(x[1]),"**\n**")

**elif** (x[0]=="R") **or** (x[0]=="r"):

cuenta -=int(x[1])

print ("retito 50 **\n**")

**elif** (x[0]=="Salir") **or** (x[0]=="salir") **or** (x[0]=="SALIR") :

print(cuenta)

**break**

**else**:

print ("Ingreso no valida**\n**")

**1.0.2 Ejercicio 2**

Escribí un programa que, dada una frase por el usuario, la muestre invertida.

[ ]: x=input("Ingresa la frase: **\n**")

y=""

**for** i **in** range(1,len(x)+1):

y+=x[-i]

y

1

**1.0.3 Ejercicio Compañero 1**

imprimir cantidad de alumnos, guardar nombre y guardar notas de alumnos y dar vuelta las notas de los alumnos

[ ]:

**1.0.4 Ejercicio compañero 2**

Crea una función que reciba una cadena de texto y que retorno 2 arreglos con minúsculas y mayús culas, cuente las letras repetidas y muestre el largo de cada lista

[ ]:

**1.0.5 Ejercicio 3**

Hacer una alarma. Para esto, se deberá fijar la hora actual y la hora deseada de la alarma. El programa deberá imprimir los minutos hasta que llegue la hora de la alarma y generará el siguiente mensaje: “Está sonando la alarma”.

[ ]: **from datetime import** datetime

**import time**

x=input("Ingrese la hora que sonara la alarma: ")

**while True**:

**try**:

time.sleep(1) *#Espera un segundo ante de volver a*␣ *,→correr el codigo*

data\_x=datetime.strptime(x,"%H:%M") *#Transformo el input a formato*␣ *,→hora y minito*

data2\_x=datetime.strftime(data\_x,"%H:%M:%S") *#Transformo el input a*␣ *,→hora, minuto y segundo*

y=time.strftime('%H:%M:%S', time.localtime()) *#Obtengo la hora*␣ *,→actual con time*

data\_y=datetime.strptime(y,"%H:%M:%S") *#Transformo esta hora*␣ *,→a fecha, ya que viene en formato String*

print ("La hora actual es:",y, ",Quedan",int((data\_x-data\_y).seconds/ *,→*60), "minutos y",((data\_x-data\_y).seconds%**60**)," segundos para que suene la␣ *,→*alarma")

**if** data2\_x==y:

print ("**\n** La alarma esta sonando")

**break**

**except ValueError**:

print("Ingre un valor valido")

**1.0.6 Ejercicio 4**

Inventario Tienda

2

Crear un programa que tenga un stock de productos almacenado en un diccionario, el diccionario tiene que contener el nombre, descripcion, stock, precio y en una lista dentro del diccionario guardar los colores o variacioens del producto, cada producto tienen que tener 3 variaciones.

Para dejar de ingresar productos, escribir la palabra SALIR.

[ ]: diccionario={}

diccionariocompleto={}

i=1

**while True**:

Nombre=input("Ingrese el nombre del producto o SALIR para dejar crear␣ *,→*producto ")

**if** Nombre=="SALIR":

**break**

Descripcion=input("Ingrese la descripción del producto:")

stock=input("Ingrese el stock del producto: ")

precio=input("Ingrese el precio del producto: ")

x=0

lista=[]

**while** x<3:

variaciones=input("Ingrese las varaciones: ")

lista.append(variaciones)

x+=1

print ("**\n**")

diccionario["Nombre"]=Nombre

diccionario["Descripcion"]=Descripcion

diccionario["stock"]=stock

diccionario["precio"]=precio

diccionario["Variaciones"]=lista

diccionariocompleto[i]=diccionario

i+=1

diccionariocompleto

[ ]: **import pandas as pd**

pd.DataFrame(diccionario)

**2 Clase 01-04-2022**

[8]: %**pwd**

[8]: 'C:\\Users\\Cristofer'

[2]: **import pandas as pd**

pd.\_\_version\_\_

3

[2]: '1.1.3'

[9]: data=pd.read\_excel("C:Downloads/bostonHousing1978.xlsx",engine="openpyxl") print (data.shape)

data.head(2)

(506, 4)

[9]: RM LSTAT PTRATIO target

0 6.575 4.98 15.3 24.0

1 6.421 9.14 17.8 21.6

[15]: data=pd.read\_csv("C:Downloads/mortgages.csv")

print (data.shape)

data.head(2)

(1080, 8)

[15]: Month Starting Balance Repayment Interest Paid Principal Paid \ 0 1 400000.00 1686.42 1000.00 686.42 1 2 399313.58 1686.42 998.28 688.14

New Balance Mortgage Name Interest Rate

0 399313.58 30 Year 0.03

1 398625.44 30 Year 0.03

[16]: data["Mortgage Name"].value\_counts()

[16]: 30 Year 720

15 Year 360

Name: Mortgage Name, dtype: int64

[17]: data["Interest Rate"].value\_counts()

[17]: 0.05 540

0.03 540

Name: Interest Rate, dtype: int64

[21]: anos\_30=data[(data["Mortgage Name"]=="30 Year") & (data["Interest Rate"]==0.03)] anos\_30

[21]: Month Starting Balance Repayment Interest Paid Principal Paid \ 0 1 400000.00 1686.42 1000.00 686.42 1 2 399313.58 1686.42 998.28 688.14 2 3 398625.44 1686.42 996.56 689.86 3 4 397935.58 1686.42 994.83 691.59 4 5 397243.99 1686.42 993.10 693.32 .. … … … … …

355 356 8364.12 1686.42 20.91 1665.51 4

356 357 6698.61 1686.42 16.74 1669.68 357 358 5028.93 1686.42 12.57 1673.85 358 359 3355.08 1686.42 8.38 1678.04 359 360 1677.04 1686.42 4.19 1682.23

New Balance Mortgage Name Interest Rate

0 399313.58 30 Year 0.03

1 398625.44 30 Year 0.03

2 397935.58 30 Year 0.03

3 397243.99 30 Year 0.03

4 396550.67 30 Year 0.03

.. … … …

355 6698.61 30 Year 0.03

356 5028.93 30 Year 0.03

357 3355.08 30 Year 0.03

358 1677.04 30 Year 0.03

359 -5.19 30 Year 0.03

[360 rows x 8 columns]

[23]: anos\_15=data[(data["Mortgage Name"]=="15 Year") & (data["Interest Rate"]==0.05)] anos\_15

[23]: Month Starting Balance Repayment Interest Paid Principal Paid \ 900 1 400000.00 3163.17 1666.66 1496.51 901 2 398503.49 3163.17 1660.43 1502.74 902 3 397000.75 3163.17 1654.16 1509.01 903 4 395491.74 3163.17 1647.88 1515.29 904 5 393976.45 3163.17 1641.56 1521.61 … … … … … …

1075 176 15619.91 3163.17 65.08 3098.09 1076 177 12521.82 3163.17 52.17 3111.00 1077 178 9410.82 3163.17 39.21 3123.96 1078 179 6286.86 3163.17 26.19 3136.98 1079 180 3149.88 3163.17 13.12 3150.05

New Balance Mortgage Name Interest Rate

900 398503.49 15 Year 0.05

901 397000.75 15 Year 0.05

902 395491.74 15 Year 0.05

903 393976.45 15 Year 0.05

904 392454.84 15 Year 0.05

… … … …

1075 12521.82 15 Year 0.05

1076 9410.82 15 Year 0.05

1077 6286.86 15 Year 0.05

1078 3149.88 15 Year 0.05

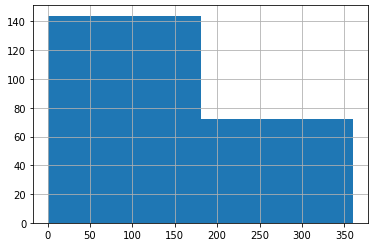
5

1079 -0.17 15 Year 0.05

[180 rows x 8 columns]

[29]: data["Month"].hist()

[29]: <AxesSubplot:>



[25]: data.describe(include="all")

[25]: Month Starting Balance Repayment Interest Paid \ count 1080.000000 1080.000000 1080.000000 1080.000000 unique NaN NaN NaN NaN top NaN NaN NaN NaN freq NaN NaN NaN NaN mean 150.500000 233244.608139 2265.486667 783.991750 std 99.544421 114887.600674 540.507327 450.838445 min 1.000000 1677.040000 1686.420000 4.190000 25% 68.000000 139742.220000 1686.420000 426.787500 50% 135.500000 249413.115000 2147.290000 761.580000 75% 225.250000 335027.117500 2762.330000 1085.610000 max 360.000000 400000.000000 3163.170000 1666.660000

Principal Paid New Balance Mortgage Name Interest Rate

count 1080.000000 1080.000000 1080 1080.000000 unique NaN NaN 2 NaN

6

top NaN NaN 30 Year NaN freq NaN NaN 720 NaN mean 1481.494917 231763.113222 NaN 0.040000 std 654.272215 115306.391990 NaN 0.010005 min 480.630000 -7.170000 NaN 0.030000 25% 914.170000 137885.062500 NaN 0.030000 50% 1392.465000 247956.530000 NaN 0.040000 75% 1967.187500 333740.135000 NaN 0.050000 max 3150.050000 399519.370000 NaN 0.050000

**2.1 Ejercicio en grupo**

Hacer un programa que pida numero de estudiantes a registrar, luego de esto pida la calificación para 4 asignaturas(matematicas, ciencias, historia, ingles), me lo van a transformar a un DF. Van a filtrar el df, con los estudiantes que hayan reprobado al menos una asignatura. La escala de 0-6 reprobado, de 7 a 10 aprobado

[91]: diccionario={}

**while True**:

**try**:

largo=int(input("Ingrese la cantidad de alumnos: "))

**for** i **in** range(0,largo):

print("**\n**")

Nombre=input("Ingrese el nombre del alumno: ")

lista=[]

**for** y **in** range(0,4):

asignatura=["matematicas", "ciencias", "historia", "ingles"]

**while True**:

print(asignatura[y])

nota=int(input(" Ingrese la nota de estudiante:" ))

**if** nota<=10 **and** nota>=0:

lista.append(nota)

**break**

**else**:

print ("Ingrese la nota nuevamente **\n**")

diccionario[Nombre]=lista

tabla=pd.DataFrame(diccionario).T

tabla.columns=asignatura

tabla["Minima nota"]=tabla.min(axis=1, numeric\_only=**True**)

tabla["Estado"]="Aprobo"

tabla["Estado"][tabla["Minima nota"]<7]="Reprobo"

7

**break**

**except ValueError**:

print("Ingrese una cantidad valida")

tabla

Ingrese la cantidad de alumnos: 2

Ingrese el nombre del alumno: Cristofer

matematicas

Ingrese la nota de estudiante:4

ciencias

Ingrese la nota de estudiante:3

historia

Ingrese la nota de estudiante:12

Ingrese la nota nuevamente

historia

Ingrese la nota de estudiante:4

ingles

Ingrese la nota de estudiante:5

Ingrese el nombre del alumno: Pablo

matematicas

Ingrese la nota de estudiante:7

ciencias

Ingrese la nota de estudiante:8

historia

Ingrese la nota de estudiante:9

ingles

Ingrese la nota de estudiante:8

<ipython-input-91-a0d1c8805532>:29: SettingWithCopyWarning: A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas docs/stable/user\_guide/indexing.html#returning-a-view-versus-a-copy tabla["Estado"][tabla["Minima nota"]<7]="Reprobo"

[91]: matematicas ciencias historia ingles Minima nota Estado Cristofer 4 3 4 5 3 Reprobo Pablo 7 8 9 8 7 Aprobo

8

**2.2**

[86]: tabla[tabla["Estado"]=="Reprobo"]

[86]: matematicas ciencias historia ingles Minima nota Estado cristofer 4 3 4 7 3 Reprobo pablo 4 5 7 2 2 Reprobo

**2.3 Ultimo ejercicio**

**Slide Type Actividad**

Crear un control de contabilidad para una tienda, en primera instancia se debe crear un constructor de inventario, donde se van a recibir productos por entrada hasta que se coloque la palabra salir. A cada uno de los productos se le va a solicitar la siguiente informacion (ID, CANTIDAD EN EXISTENCIA, CANTIDAD VENDIDA, ITEMS PERDIDOS, TOTAL GANANCIA). Una vez que finaliza el constructor de inventario, se debe mostrar el dataframe construido y un menu con 5 filtros diferentes a implementar que seran elegidos con los numeros del 1 al 5, cuando se digite 0, el programa termina.

[ ]: diccionario={}

diccionario\_completo={}

**while True**:

ID=input("**\n** Ingrense el ID o Salir: ")

**if** ID=="Salir":

**break**

CANTIDAD=input("Ingrense la cantidad del producto: ")

CANTIDAD\_PERDIDA=input("Ingrense la cantidad perdida: ")

ITEMS=input("Ingrense los items perdidos:" )

GANANCIA=input("Ingrense la ganancia")

diccionario["Cantidad"]=CANTIDAD

diccionario["Cantidad\_perdida"]=CANTIDAD\_PERDIDA

diccionario["Items"]=ITEMS

diccionario["Ganancia"]=GANANCIA

diccionario\_completo[ID]=diccionario

tabla=pd.DataFrame(diccionario\_completo).T

tabla["Cantidad"]=pd.to\_numeric(tabla["Cantidad"])

tabla["Cantidad\_perdida"]=pd.to\_numeric(tabla["Cantidad\_perdida"]) tabla["Items"]=pd.to\_numeric(tabla["Items"])

tabla["Ganancia"]=pd.to\_numeric(tabla["Ganancia"])

print ("**\n**",tabla)

**while True**:

print ('**\n** INGRESE UNA OPCION DE 1 A 5',"**\n**", '**\033**[1m'+'Numero 1: *,→*'+'**\033**[0m',"Obtener items con Ganancia positiva **\n**",'**\033**[1m'+"Numero 2: *,→*"+'**\033**[0m' ,"Obtener item con perdidas monetarias")

9

print ('**\033**[1m'+ " Numero 3:"+'**\033**[0m' ,"Para ver la cantidad perdida de␣ *,→*los items **\n**", '**\033**[1m'+"Numero 4:"+'**\033**[0m'," Para obtener cuantos items␣ *,→*hay **\n**",'**\033**[1m'+"Numero 5:" +'**\033**[0m', "Ganacia total") print ('**\033**[1m'+'Para salir solo ingrese 0'+'**\033**[0m')

**while True**:

print ("**\n**")

opcion=int(input("Ingrese un valor valido: "))

**if** opcion==0:

print ("El menu desplegable ha finalizado")

**break**

**elif** opcion <0 **and** opcion>5:

print("Ingrese una opcion validad")

**if** opcion==1:

print(tabla[tabla["Ganancia"]>0])

**if** opcion==2:

print(tabla[tabla["Ganancia"]<0])

**if** opcion==3:

print(int(tabla["Cantidad\_perdida"].sum()))

**if** opcion==4:

print(int(tabla["Items"].count()))

**if** opcion==5:

print(int(tabla["Ganancia"].sum()))

Ingrense el ID o Salir: 45345

Ingrense la cantidad del producto: 234

Ingrense la cantidad perdida: 234

Ingrense los items perdidos:234

Ingrense la ganancia234

Ingrense el ID o Salir: 34

Ingrense la cantidad del producto: 234

Ingrense la cantidad perdida: 234

Ingrense los items perdidos:234

Ingrense la ganancia234

Ingrense el ID o Salir: Salir

Cantidad Cantidad\_perdida Items Ganancia

45345 234 234 234 234

34 234 234 234 234

INGRESE UNA OPCION DE 1 A 5

10

**Numero 1:** Obtener items con Ganancia positiva **Numero 2:** Obtener item con perdidas monetarias **Numero 3:** Para ver la cantidad perdida de los items **Numero 4:** Para obtener cuantos items hay **Numero 5:** Ganacia total

**Para salir solo ingrese 0**

Ingrese un valor valido: 1

si

Cantidad Cantidad\_perdida Items Ganancia 45345 234 234 234 234 34 234 234 234 234

Ingrese un valor valido: 2

Empty DataFrame

Columns: [Cantidad, Cantidad\_perdida, Items, Ganancia] Index: []

Ingrese un valor valido: 3

468

Ingrese un valor valido: 4

2

Ingrese un valor valido: 5

468

11