UNIVERSIDAD DE PIURA

FACULTAD DE INGENIERÍA

PYTHON PARA EL ANÁLISIS DE DATOS 1

Informe Final

Equipo 11

Gutierrez Lozano, Gonzalo Raúl
Soto Rodríguez, Fernando Daniel
Torres Cabrera, Alisson Xiomara
Vega Sanz, Victor Alejandro
Villalobos Feria, David Sebastian

Profesor:
Ing. Pedro Rotta.

Lima, enero 2021

ÍNDICE

1. INTRODUCCIÓN

1.1. Problemática

En un salón de clases siempre se tendrá la data de los alumnos que el profesor pueda recolectar, y por lo general, tendrá anotada la información en un registro físico o virtual. Pero tener esta información sin procesar sería una pérdida de tiempo. Es por ello que para un usuario (profesor o alumno) sería de utilidad contar con programas que extraigan estos datos y realice más funcionalidades, esto contribuirá a un mejor análisis de los datos y en consecuencia a una mejor toma de decisiones en beneficio de los alumnos y profesores, en especial si el código desarrollado hace uso de poco espacio y tiene mayor velocidad.

1.2. Resolución

Para el análisis de los datos, los programas servirán como un asistente que ofrecerá funciones, mostrará gráficas, y ayudará en la interpretación de los mismos; mientras que para que sea más ligero y rápido se hará uso de algunas librerías.

A partir de una lista de alumnos, se realizará las siguientes funciones:

- Permitirá ingresar filas (datos de nuevos alumnos) o columnas de datos (una nueva característica de un alumno).
- Permitirá filtrar filas según una característica (Edad, Cursos, Promedio, etc.).
- Mostrará gráficas que muestren más a detalle cómo se está desempeñando cada alumno.

2. ANÁLISIS DEL SISTEMA

En este apartado explicaremos con mayor detalle nuestras líneas de código.

Programa 1:

```
[ ]  #Importamos las librerias necesarias
```

```
[ ]  import numpy as np
     import pandas as pd
     import seaborn as sns
     from pandas import DataFrame
     import matplotlib.pyplot as plt
     from matplotlib.pylab import subplots
```

```
[ ]  #Subimos el excel
```

```
[ ]  ruta_alumnos="/content/drive/MyDrive/Trabajo python/Alumnos.xlsx"
     alumnos_df=pd.read_excel(ruta_alumnos)
     alumnos_df.head()
     alumnos_df
```

Descripción:

Importamos las librerías que necesitaremos para la actualización y análisis de los datos, de la base de datos del documento excel que subimos.

Programa 2:

```
[ ]  #Ingresamos datos a la base de datos para que se actualicen
```

```
[ ]  def read():
       global ruta_alumnos
       df= pd.read_excel(ruta_alumnos)
       return df

     def save(df):
       global ruta_alumnos
       df.to_excel(ruta_alumnos)
```

Descripción:

La función read() leerá el archivo .xlsx y devolverá este como data frame(df).

La función save(df) guardará en el mismo archivo excel el data frame que recibe como entrada.

En ambos casos aparece la variable global ruta_alumnos ya que en el siguiente apartado se le asignará un valor (una ruta) que está fuera de estas funciones.

Programa 3:

```
def Columns (df):
  columnas =df.columns
  list_columns=list(columnas)
  return list_columns

ruta_alumnos="/content/drive/MyDrive/Trabajo python/Alumnos.xlsx"
def añadir_alumno():
  df= read()
  dicc_fila={}
  columnas_df=Columns(df)
  for i in columnas_df:
    if df[i].dtype==object:
     dicc_fila[i]=input("Ingrese el valor de columna "+i+": ")
    elif df[i].dtype==int:
     dicc_fila[i]=int(input("Ingrese el valor de columna "+i+": "))
  df=df.append(dicc_fila,ignore_index=True)
  df_reset=df.set_index('Alumnos')
  save(df_reset)

x = input("Ingrese S si quiere ingresar y otra tecla para cerrar: ")
while x=="S":
  añadir_alumno()
  x = input("Ingrese S si quiere ingresar y otra tecla para cerrar: ")
print("ya no hay más datos")
```

Descripción:

Luego de definir las funciones que leen y guardan el archivo Excel, en este apartado se añadirán nuevas filas con información de nuevos alumnos que será guardada en el mismo archivo incluso si el programa se vuelve a ejecutar.

Primero el usuario debe ingresar “S” en el terminal, mientras que se cumpla esto el programa ejecutará la función anadir_alumno(); en este la función read() leerá el archivo, el diccionario creado será donde se almacenará cada valor ingresado por el usuario en su respectiva columna. Esto se logra usando el bucle "for", ya sea que el tipo de dato que almacena una columna es String o entero, el programa lo guardará como tal en el archivo mediante la función save() definida anteriormente.

Cabe resaltar que antes de que la información se guarde en el archivo, se debe eliminar la columna de índices que genera automáticamente un dataframe luego de haber sido leído.

Así sucesivamente el usuario podrá seguir añadiendo las filas que desee mientras que introduzca la letra "S", de lo contrario se le indicará que ya no hay más datos por registrar.

Programa 4, 5:

```
[ ]  #Filtramos datos por medio de la característica de la tabla
     df=read()
     respuesta=df.loc[df["Edad"]==18]
     print(respuesta)
```

```
              Alumnos       Curso  Promedio  Edad
1    Domingo Arévalo  Matemática         4    18
10  Maricarmen Tapia      Python        13    18
```

```
[ ]  #Realizamos un histograma para el analizar las edades de los alumnos
```

```
[ ]  style=dict(size=12,color="black",fontfamily="monospace",fontstyle="oblique")
     plt.style.use('fivethirtyeight')
     alumnos_lista = list(df["Edad"])
     plt.hist(alumnos_lista, bins = 6, alpha = 0.5, histtype='bar', color='steelblue', edgecolor = 'none')
     plt.text(19.3,4,"Punto Máx",**style)
     plt.show()
```

Descripción:

Mediante el primer código podremos filtrar los contenidos de la tabla por los títulos de cada columna y sus datos. Y con el segundo código podemos realizar una histograma con las edades de los alumnos y señalar cuál es la edad más común entre los mismos.

Programa 6, 7 y 8:

```
[ ] lista_columnas_df=Columns(df)
    print(lista_columnas_df)
```

```
['Alumnos', 'Curso', 'Promedio', 'Edad']
```

```
[ ] len(df["Alumnos"])
```

```
11
```

```
[ ] #Generar nuevas características para los datos. (Nuevas columnas)
    horarios=[]
    for i in range(0,len(df["Alumnos"])):
      horas_alum_estudio=int(input("Ingrese las horas de estudio del alumno: "))
      horarios.append(horas_alum_estudio)

    df.insert(4, "Horas Estudiadas x Semana", horarios, allow_duplicates=False)
    df
```

Descripción:

Con el primer código, podemos observar los títulos que tienen las columnas de nuestro Data Frame y saber también cuantas columnas tenemos. Mientras que con el segundo código, determinamos la extensión de datos que tiene cada columna y por ende la cantidad de filas contenidas en nuestra tabla.

Con el tercer código y habiendo recopilado la información de los dos códigos anteriores, mediante un bucle For el usuario puede colocar los datos de la nueva columna, que para el caso práctico es la columna Horas Estudiadas x Semana, que se colocará con el método insert.

Programa 9:

```
df.insert(3, "Máx Calificación", 20, allow_duplicates=False)
df
```

Descripción:

Mediante el método insert, colocamos otra columna llamada “Máx Calificación”, que más adelante nos servirá para poder determinar el rendimiento de los estudiantes en sus respectivas clases.

Programa 10:

```
df["Rendimiento"]= (df["Promedio"]/df["Máx Calificación"])
df
```

Descripción:

Con el siguiente programa, operando la columna Promedio/Máx Calificación, determinamos los datos que tendrá la nueva columna, que mostrará el rendimiento de los estudiantes.

Programa 11:

```
#Análisis de datos

sns.relplot(x="Horas Estudiadas x Semana",y="Promedio",data=df)
plt.show()
```

Descripción:

Mediante el siguiente programa realizamos una gráfica donde el eje X son los datos de la columna "Horas Estudiadas x Semana" y el eje Y el "Promedio", de modo que podemos determinar la relación que hay entre las horas de estudio que tienen a la semana los estudiantes y sus calificaciones finales en los cursos.

Programa 12:

```
sns.relplot(x="Horas Estudiadas x Semana",y="Promedio",hue="Edad",data=df)
plt.show()
```

Descripción:

Con el siguiente programa podemos determinar respecto a la relación de "Horas Estudiadas x Semana" y "Promedio", con que edad tienen esos alumnos que obtuvieron esos resultados.

Programa 13:

```
sns.relplot(x="Edad",y="Rendimiento",data=df)
plt.show()
```

Descripción:

Con este programa podemos determinar la relación que hay entre la edad de los alumnos y el rendimiento que tuvieron en el curso.

3. EJEMPLOS AL CORRER EL PROGRAMA

Programa 1: Importación de librerías y carga del documento Excel.

	Alumnos	Curso	Promedio	Edad
0	Juan Arbulú	Matemática	11	20
1	Domingo Arévalo	Matemática	4	18
2	Pedro Flores	Matemática	19	19
3	Enrique Villaseca	Python	13	26
4	Roberto Morales	Python	8	20
5	Patricio Fernandez	Lenguaje	11	21
6	Jorge Bazán	Lenguaje	13	22
7	Ernesto Villaseca	Lenguaje	10	24
8	Fernando Guzmán	Lenguaje	20	20

Programa 2, 3: Ingreso de datos a la base de datos para que se actualice.

```
Ingrese S si quiere ingresar y otra tecla para cerrar: S
Ingrese el valor de columna Alumnos: Juan Almara
Ingrese el valor de columna Curso: Python
Ingrese el valor de columna Promedio: 15
Ingrese el valor de columna Edad: 20
Ingrese S si quiere ingresar y otra tecla para cerrar: S
Ingrese el valor de columna Alumnos: Maricarmen Tapia
Ingrese el valor de columna Curso: Python
Ingrese el valor de columna Promedio: 13
Ingrese el valor de columna Edad: 18
Ingrese S si quiere ingresar y otra tecla para cerrar: x
ya no hay más datos
```

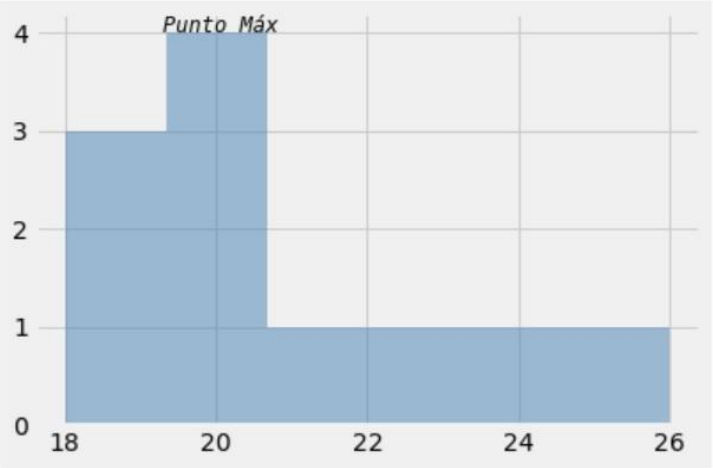
	Alumnos	Curso	Promedio	Edad
0	Juan Arbulú	Matemática	11	20
1	Domingo Arévalo	Matemática	4	18
2	Pedro Flores	Matemática	19	19
3	Enrique Villaseca	Python	13	26
4	Roberto Morales	Python	8	20
5	Patricio Fernandez	Lenguaje	11	21
6	Jorge Bazán	Lenguaje	13	22
7	Ernesto Villaseca	Lenguaje	10	24
8	Fernando Guzmán	Lenguaje	20	20
9	Juan Almara	Python	15	20
10	Maricarmen Tapia	Python	13	18

Programa 4: Filtramos datos por medio de la característica de la tabla.

```
             Alumnos       Curso  Promedio  Edad
1    Domingo Arévalo  Matemática         4    18
10  Maricarmen Tapia      Python        13    18
```

```
Ingrese las horas de estudio del alumno: 6
Ingrese las horas de estudio del alumno: 3
Ingrese las horas de estudio del alumno: 12
Ingrese las horas de estudio del alumno: 7
Ingrese las horas de estudio del alumno: 5
Ingrese las horas de estudio del alumno: 7
Ingrese las horas de estudio del alumno: 7
Ingrese las horas de estudio del alumno: 6
Ingrese las horas de estudio del alumno: 12
Ingrese las horas de estudio del alumno: 9
Ingrese las horas de estudio del alumno: 8
```

Programa 5: Realizamos un histograma para analizar las edades de los alumnos.



Programa 6, 7 y 8: Generamos nuevas características para los datos. (Nuevas columnas)

```
Ingrese las horas de estudio del alumno: 6
Ingrese las horas de estudio del alumno: 3
Ingrese las horas de estudio del alumno: 12
Ingrese las horas de estudio del alumno: 7
Ingrese las horas de estudio del alumno: 5
Ingrese las horas de estudio del alumno: 7
Ingrese las horas de estudio del alumno: 7
Ingrese las horas de estudio del alumno: 6
Ingrese las horas de estudio del alumno: 12
Ingrese las horas de estudio del alumno: 9
Ingrese las horas de estudio del alumno: 8
```

"Horas Estudiadas x Semana"

	Alumnos	Curso	Promedio	Edad	Horas Estudiadas x Semana
0	Juan Arbulú	Matemática	11	20	6
1	Domingo Arévalo	Matemática	4	18	3
2	Pedro Flores	Matemática	19	19	12
3	Enrique Villaseca	Python	13	26	7
4	Roberto Morales	Python	8	20	5
5	Patricio Fernandez	Lenguaje	11	21	7
6	Jorge Bazán	Lenguaje	13	22	7
7	Ernesto Villaseca	Lenguaje	10	24	6
8	Fernando Guzmán	Lenguaje	20	20	12
9	Juan Almara	Python	15	20	9
10	Maricarmen Tapia	Python	13	18	8

Programa 9: "Máx Calificación"

	Alumnos	Curso	Promedio	Máx Calificación	Edad	Horas Estudiadas x Semana
0	Juan Arbulú	Matemática	11	20	20	6
1	Domingo Arévalo	Matemática	4	20	18	3
2	Pedro Flores	Matemática	19	20	19	12
3	Enrique Villaseca	Python	13	20	26	7
4	Roberto Morales	Python	8	20	20	5
5	Patricio Fernandez	Lenguaje	11	20	21	7
6	Jorge Bazán	Lenguaje	13	20	22	7
7	Ernesto Villaseca	Lenguaje	10	20	24	6
8	Fernando Guzmán	Lenguaje	20	20	20	12
9	Juan Almara	Python	15	20	20	9
10	Maricarmen Tapia	Python	13	20	18	8

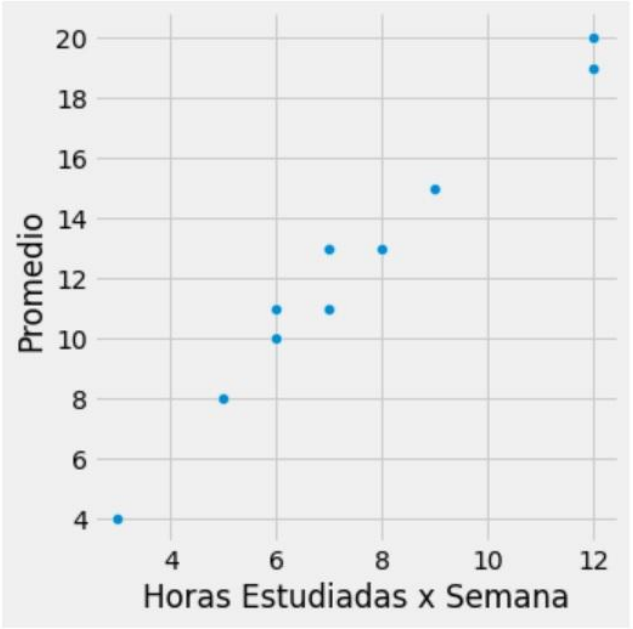
Programa 10: "Rendimiento"

	Alumnos	Curso	Promedio	Máx Calificación	Edad	Horas Estudiadas x Semana	Rendimiento
0	Juan Arbulú	Matemática	11	20	20	6	0.55
1	Domingo Arévalo	Matemática	4	20	18	3	0.20
2	Pedro Flores	Matemática	19	20	19	12	0.95
3	Enrique Villaseca	Python	13	20	26	7	0.65
4	Roberto Morales	Python	8	20	20	5	0.40
5	Patricio Fernandez	Lenguaje	11	20	21	7	0.55
6	Jorge Bazán	Lenguaje	13	20	22	7	0.65
7	Ernesto Villaseca	Lenguaje	10	20	24	6	0.50
8	Fernando Guzmán	Lenguaje	20	20	20	12	1.00
9	Juan Almara	Python	15	20	20	9	0.75
10	Maricarmen Tapia	Python	13	20	18	8	0.65

Análisis de datos.

Programa 11:

Variables = **"Horas Estudiadas x Semana"** y **"Promedio"**.

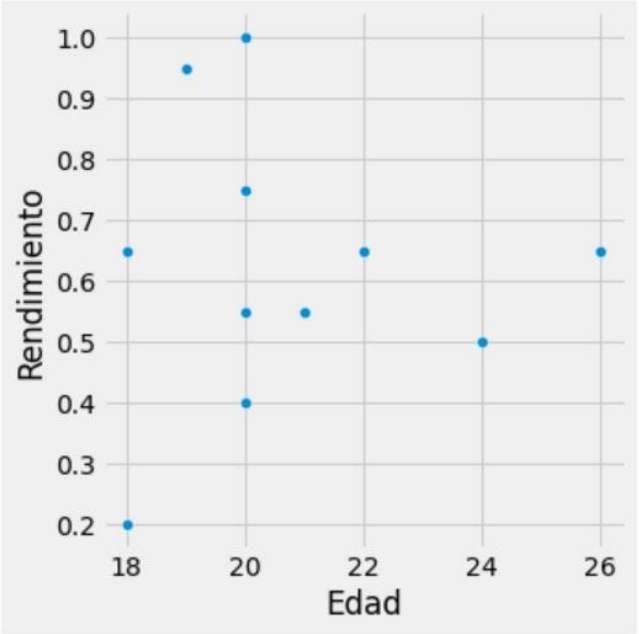


Programa 12:

Variables = **"Horas Estudiadas x Semana"**, **"Promedio"** y **"Edad"**.

Programa 13:

Variables = **"Edad"** y **"Rendimiento"**



4. CONCLUSIONES

- El programa resulta ser más intuitivo con el usuario gracias a funciones como input o print. Con el primero puede indicarle al usuario qué o cómo debe ingresar la información y con el segundo le señalaría en caso el programa terminara de registrar datos.
- Es conveniente definir funciones cuando se hace notorio que se tiene que repetir la misma operación a lo largo del código. Esto simplifica muchos problemas y el programa no llega a ser muy pesado.
- Las librerías importadas ayudaron a que el programa sea óptimo en velocidad y peso.
- El programa muestra gráficas mediante las cuales el usuario, si es el maestro, puede darse cuenta del progreso que tienen sus estudiantes e incentivar más a los que les está yendo mal. En el caso que el usuario sea alumno, este puede ver su rendimiento respecto a sus compañeros y observando a los estudiantes que les va mejor, pueden darse una idea de cuántas horas tiene que estudiar uno para obtener buenos resultados.
- Mediante más datos obtengamos, los resultados y por lo tanto la toma de decisiones que puedan tomar los usuarios, será más precisa.