
 UNSA UNIVERSIDAD NACIONAL DE SAN AGUSTÍN DE AREQUIPA	ESCUELA PROFESIONAL DE INGENIERÍA DE TELECOMUNICACIONES LABORATORIO DE INFORMÁTICA EXPERIENCIA: 05 IMPORTAR FICHEROS EXCEL CON PANDAS	Emisión: 28/04/2022 Página 1 / 17	
--	--	--	--

ESCUELA PROFESIONAL DE INGENIERÍA DE TELECOMUNICACIONES



LABORATORIO DE: COMPUTACIÓN 2

EXPERIENCIA N°: 05

TÍTULO DE LA EXPERIENCIA:

IMPORTAR FICHEROS EXCEL CON PANDAS

Alumno(os):		Grupal	Indiv.	Total
1. Prieto Tito Manuel Ismael			x	
2.				
3.				
4.				
Grupo:	b	Docente: Ing William Mullisaca Atamari		
Semestre:	3			
Fecha entrega:	de 15-06-22	Hora:	12.00	

 UNSA UNIVERSIDAD NACIONAL DE SAN AGUSTÍN DE AREQUIPA	ESCUELA PROFESIONAL DE INGENIERÍA DE TELECOMUNICACIONES LABORATORIO DE INFORMÁTICA EXPERIENCIA: 05 IMPORTAR FICHEROS EXCEL CON PANDAS	Emisión: 28/04/2022 Página 2 / 17	
--	--	--	--

LABORATORIO N° 05

IMPORTAR FICHEROS EXCEL CON PANDAS

I.- OBJETIVOS:

-. Que el estudiante sea capaz de importar ficheros Excel con pandas en Python.

II.- FUNDAMENTO TEÓRICO:

2.1. Programando con python como lenguaje de programación hay muchas librerías muy útiles para abrir excel: Openpyxl, XlsxWriter, y Pandas

En esta guía utilizaremos Pandas debido a su flexibilidad y cantidad de herramientas. Mi recomendación es: utilizar Pandas si van a trabajar con muchos datos (porque tiene muchas herramientas) y aprovechar las otras librerías si quieren hacer desarrollo y solo quieren guardar datos en un archivo excel.

Para esta guía utilizaremos un set de datos : Austin Weather

Primero importaremos las librerías necesarias

```
import pandas as pd
```

Los archivos están guardados en la misma carpeta que el dónde estoy ejecutando el código, por eso no es necesario colocar carpetas o subcarpetas en la dirección.

```
ipath = 'austion_weather.xlsx'
df = pd.read_excel(ipath)
df.head()
```

Si ejecutan esto les debería mostrar las primeras 5 filas del archivo excel. Está bastante feo porque solo la primera columna tiene datos.



	Content	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unnamed: 6	Unnamed: 7	Unnamed: 8
0	Contains the:	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	Date (YYYY-MM-DD)	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	TempHighF (High temperature, in Fahrenheit)	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	TempAvgF (Average temperature, in Fahrenheit)	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Vamos a saltarnos unas cuantas filas para encontrar los datos.

```
df = pd.read_excel(ipath, skiprows=26)
df.head()
```

	Date	TempHighF	TempAvgF	TempLowF	DewPointHighF	DewPointAvgF	DewPointLowF	Hum
0	2013-12-21	74	60	45	67	49	43	93
1	2013-12-22	56	48	39	43	36	28	93
2	2013-12-23	58	45	32	31	27	23	76
3	2013-12-24	61	46	31	36	28	21	89
4	2013-12-25	58	50	41	44	40	36	86

5 rows × 9 columns

¿Eso está mejor verdad? Ahora pandas me muestra los títulos de la columna como ser: Date, TempAvgF, TempLowF, etc



Ahora veamos algo que no se ve a simple vista. ¿Cómo son los datos que tenemos? Para eso ejecutamos:

```
df.info()
```

Esto nos debería mostrar los detalles por cada columna:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1319 entries, 0 to 1318
Data columns (total 21 columns):
Date                                1319 non-null object
TempHighF                          1319 non-null int64
TempAvgF                           1319 non-null int64
TempLowF                           1319 non-null int64
DewPointHighF                      1319 non-null object
DewPointAvgF                       1319 non-null object
DewPointLowF                       1319 non-null object
HumidityHighPercent                1319 non-null object
HumidityAvgPercent                 1319 non-null object
HumidityLowPercent                 1319 non-null object
SeaLevelPressureHighInches         1319 non-null object
SeaLevelPressureAvgInches          1319 non-null object
SeaLevelPressureLowInches          1319 non-null object
VisibilityHighMiles                1319 non-null object
VisibilityAvgMiles                 1319 non-null object
VisibilityLowMiles                 1319 non-null object
WindHighMPH                       1319 non-null object
```



```
WindAvgMPH          1319 non-null object
WindGustMPH         1319 non-null object
PrecipitationSumInches 1319 non-null object
Events              1319 non-null object
dtypes: int64(3), object(18)
memory usage: 216.5+ KB
```

¿Qué leemos? Resumiendo: cantidad de datos por columna, qué tipo de datos es (object, int64) Object es una mezcla de datos (texto, números, etc. Pandas llama así a las columnas con datos mixtos) y el int64, se refiere a números enteros.

Hay más información, pero por ahora esto es todo lo que nos interesa.

Una tabla como está nos puede dar problemas para trabajar. ¿Por qué? Porque por ejemplo no puedo sacar datos de estadísticos con columnas que tienen datos de texto y números. No puedo sacar promedios si Pandas no reconoce a los datos como números.

¿Qué ocurre aquí? Por qué Pandas me dice que la columna de DewPointAvgF tiene texto y números. Si revisan esa columna verán que cuando no hay datos no dejan la celda vacía, colocan un guión (-), un ejemplo es el 14-06-2014 o el 10-08-2015. Esto hace que Pandas crea que hay texto y número... y tiene razón!!!



Vamos a avisarle a Pandas que los guiones significan datos inexistentes. Eso se llama NaN Values, y Pandas si reconoce esos datos, los considera datos inexistentes.

```
df = pd.read_excel(ipath, skiprows=26, na_values='-')
df.info()

<class 'pandas.core.frame.DataFrame'>
```



```
RangeIndex: 1319 entries, 0 to 1318
Data columns (total 21 columns):
Date                                1319 non-null object
TempHighF                          1319 non-null int64
TempAvgF                           1319 non-null int64
TempLowF                           1319 non-null int64
DewPointHighF                      1312 non-null float64
DewPointAvgF                       1312 non-null float64
DewPointLowF                       1312 non-null float64
HumidityHighPercent                1317 non-null float64
HumidityAvgPercent                 1317 non-null float64
HumidityLowPercent                 1317 non-null float64
SeaLevelPressureHighInches         1316 non-null float64
SeaLevelPressureAvgInches          1316 non-null float64
SeaLevelPressureLowInches          1316 non-null float64
VisibilityHighMiles                1307 non-null float64
VisibilityAvgMiles                 1307 non-null float64
VisibilityLowMiles                 1307 non-null float64
WindHighMPH                        1317 non-null float64
WindAvgMPH                        1317 non-null float64
WindGustMPH                        1315 non-null float64
PrecipitationSumInches             1319 non-null object
Events                            1319 non-null object
dtypes: float64(15), int64(3), object(3)
memory usage: 216.5+ KB
```

 UNSA UNIVERSIDAD NACIONAL DE SAN AGUSTÍN DE AREQUIPA	ESCUELA PROFESIONAL DE INGENIERÍA DE TELECOMUNICACIONES LABORATORIO DE INFORMÁTICA EXPERIENCIA: 05 IMPORTAR FICHEROS EXCEL CON PANDAS	Emisión: 28/04/2022 Página 7 / 17	
--	---	--	--

Cambio, ¿verdad? Ahora Pandas me da nuevos tipos de datos. Los float64, que significan que son datos con comas, o sea, decimales.

Las últimas dos columnas continúan siendo object, debido a que PrecipitationSumInches tiene un valor de trazas (T). Cada vez que precipito/llovió menos de 0.01 inches (pulgadas) colocaron T en la celda. No es un dato inexistente, y debe ser tratado de una forma diferente a la anterior.

La columna Events tiene descripciones de los eventos, podemos dejarlo en texto.

Bien... En próximos tutoriales veremos cómo convertir la columna Date en tipo DateTime, esto nos permitira trabajar estos datos como una serie de tiempo.

Cualquier duda pueden dejarla en los comentarios, o sino buscarme en las redes sociales.

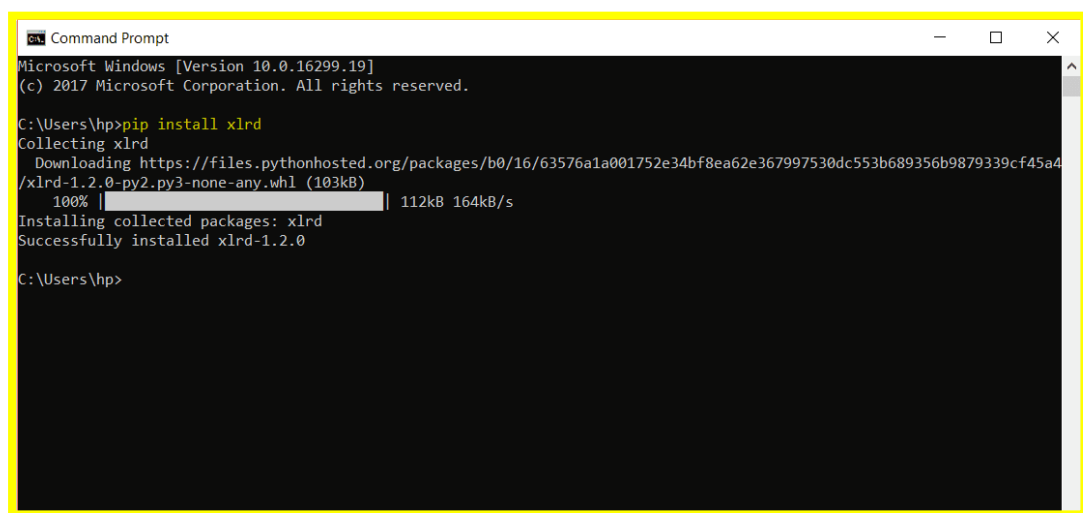
Nota: Las imagenes son del Jupyter Notebook. Más adelante hablaré de esa herramienta, pero si ejecutan los codigos en algun IDE (Spyder, Pycharm, etc.) o de la consola interactiva de Python no deberían tener problemas en obtener los mismos resultados.

2.2. LEER UN ARCHIVO DE EXCEL

Puedes leer desde un archivo de Excel usando el método `read_excel()` de pandas. Para esto, necesitas importar un módulo más llamado `xlrd`.

Instala `xlrd` usando pip:

```
pip install xlrd
```



```

Microsoft Windows [Version 10.0.16299.19]
(c) 2017 Microsoft Corporation. All rights reserved.

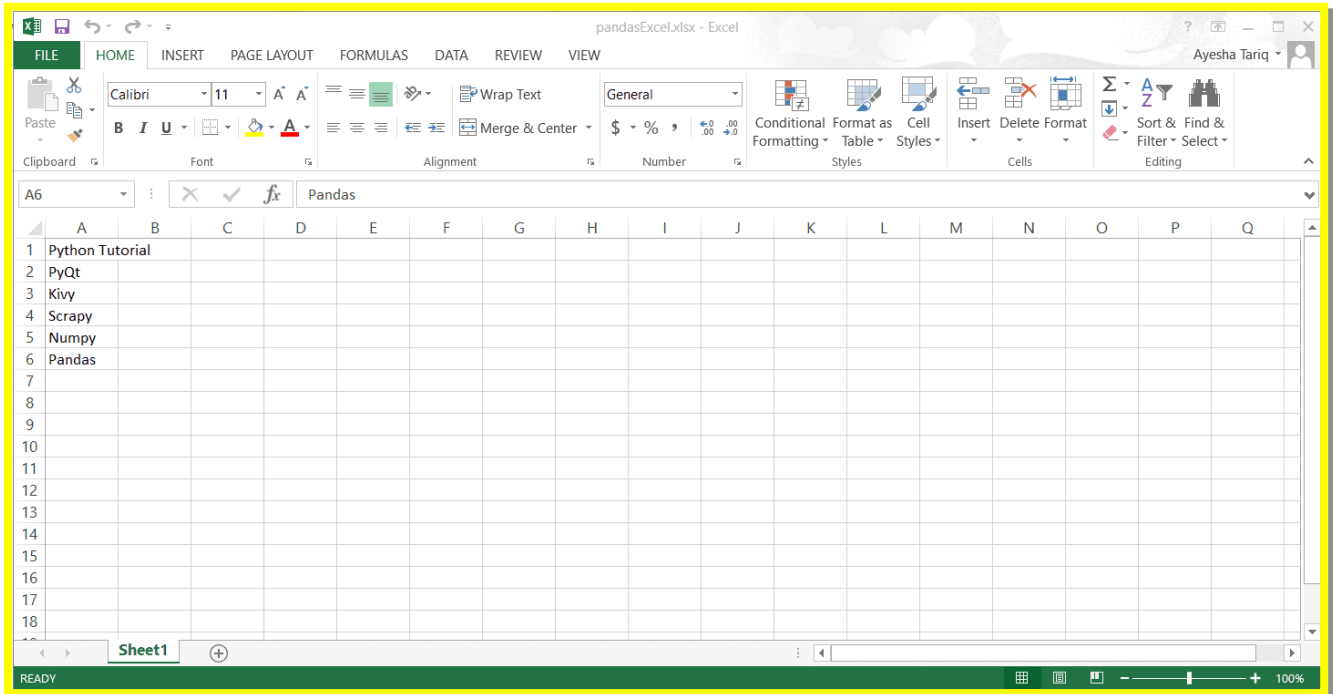
C:\Users\hp>pip install xlrd
Collecting xlrd
  Downloading https://files.pythonhosted.org/packages/b0/16/63576a1a001752e34bf8ea62e367997530dc553b689356b9879339cf45a4/xlrd-1.2.0-py2.py3-none-any.whl (103kB)
    100% |#####| 112kB 164kB/s
Installing collected packages: xlrd
Successfully installed xlrd-1.2.0

C:\Users\hp>
  
```

El siguiente ejemplo muestra cómo leer de una hoja de Excel:



1. Creamos una hoja de Excel con los siguientes contenidos:



2. Importa el módulo de pandas.



```
import pandas
```

3. Pasaremos el nombre del archivo de Excel y el número de hoja del que necesitamos leer los datos al método read_excel ().

```
pandas.read_excel('pandasExcel.xlsx', 'Sheet1')
```

El fragmento anterior generará el siguiente resultado:

```
>>> import pandas
>>> pandas.read_excel('pandasExcel.xlsx', 'Sheet1')
Python Tutorial
0          PyQt
1          Kivy
2        Scrappy
3         Numpy
4         Pandas
>>>
```


 UNSA <small>UNIVERSIDAD NACIONAL DE SAN AGUSTÍN DE AREQUIPA</small>	ESCUELA PROFESIONAL DE INGENIERÍA DE TELECOMUNICACIONES LABORATORIO DE INFORMÁTICA EXPERIENCIA: 05 IMPORTAR FICHEROS EXCEL CON PANDAS		Emisión: 28/04/2022 Página 9 / 17	
---	--	--	--	--

Si verificas el tipo de salida usando la palabra clave de type, te dará el siguiente resultado:

```
<class 'pandas.core.frame.DataFrame'>
```

Este resultado es llamado **DataFrame**. Esa es la unidad básica de pandas con la que se va a tratar mas adelante.

El DataFrame es una estructura de 2 dimensiones etiquetada donde podemos almacenar datos de diferentes tipos. DataFrame es similar a una tabla SQL o una hoja de cálculo de Excel.

2.3. IMPORTAR ARCHIVO CSV

Para leer un archivo CSV, puedes usar el método read_csv () de pandas.

Importa el módulo de pandas:

```
import pandas
```

Ahora llama al método read_csv () de la siguiente manera:

```
pandas.read_csv('Book1.csv')
```

Book1.csv tiene el siguiente contenido:



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	First Name	Last Name	Age														
2	Jean	Rukebesha	22														
3	Susan	Anthony	17														
4	Jason	James	18														
5																	
6																	
7																	
8																	
9																	
10																	
11																	
12																	
13																	
14																	
15																	
16																	
17																	
18																	

El código generará el siguiente DataFrame:

```
>>> pandas.read_csv('Book1.csv')
  First Name  Last Name  Age
0      Jean  Rukebesha   22
1     Susan   Anthony   17
2     Jason     James   18
>>>
```

III.- RECOMENDACIONES EN SEGURIDAD

3.1. En condiciones de una emergencia Identifique:

- Vías de acceso y evacuación
- Equipos de respuesta a emergencias
- Señalización de seguridad

3.2. Complete el ATS (Anexo 1) y cumpla las condiciones obligatorias para el uso del ambiente

IV.- EQUIPOS Y MATERIALES A UTILIZAR:

Cantidad	Descripción
1	PC con software Python

V.- PROCEDIMIENTO:

5.1. Ejecutar, el ítem 2.1 con el archivo Excel que te da el docente.

```
In [2]: import pandas as pd
ipath = 'austin_weather.xlsx'
df = pd.read_excel(ipath)
df.head()
```

```
Out[2]:
```

	Date	TempHighF	TempAvgF	TempLowF	DewPointHighF	DewPointAvgF	DewPointLowF	HumidityHighPercent	HumidityAvgPercent	HumidityLowPercent
0	2013-12-21	74	60	45	67	49	43	93	75	58
1	2013-12-22	56	48	39	43	36	28	93	68	44
2	2013-12-23	58	45	32	31	27	23	76	52	42
3	2013-12-24	61	46	31	36	28	21	89	56	42
4	2013-12-25	58	50	41	44	40	36	86	71	58

5 rows x 21 columns

```
In [3]: df = pd.read_excel(ipath, skiprows=26)
df.head()
```

```
Out[3]:
```

	2014-01-15	64	52	40	29	23	16	49	35	21	...	30.33	30.17	10	10.1	10.2	14	4	21.1	0
0	2014-01-16	72	56	40	31	27	23	55	38	20	...	30.13	30.06	10	10	10	14	4	20	0
1	2014-01-17	64	54	43	31	20	15	45	31	17	...	30.29	30.21	10	10	9	14	5	21	0
2	2014-01-18	70	54	37	41	33	23	59	45	31	...	30.19	30.1	10	10	10	12	2	19	0
3	2014-01-19	72	59	45	41	30	18	64	39	14	...	30.22	30.12	10	10	10	9	3	16	0
4	2014-01-20	82	64	46	51	36	23	71	44	17	...	30	29.85	10	10	10	16	5	26	0

5 rows x 21 columns

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1293 entries, 0 to 1292
Data columns (total 21 columns):
#   Column      Non-Null Count  Dtype
---  -
0    2014-01-15  1293 non-null  object
1    64          1293 non-null  int64
2    52          1293 non-null  int64
3    40          1293 non-null  int64
4    29          1293 non-null  object
5    23          1293 non-null  object
6    16          1293 non-null  object
7    49          1293 non-null  object
8    35          1293 non-null  object
9    21          1293 non-null  object
10   30.47       1293 non-null  object
11   30.33       1293 non-null  object
12   30.17       1293 non-null  object
13   10          1293 non-null  object
14   10.1        1293 non-null  object
15   10.2        1293 non-null  object
16   14          1293 non-null  object
17   4           1293 non-null  object
```



```

In [10]: df
Out[10]:
class 'pandas.core.frame.DataFrame'>
RangeIndex: 1293 entries, 0 to 1292
Data columns (total 21 columns):
#   Column              Non-Null Count  Dtype
---  ---
0   2014-01-15          1293 non-null   object
1   64                  1293 non-null   int64
2   52                  1293 non-null   int64
3   40                  1293 non-null   int64
4   29                  1286 non-null   float64
5   23                  1286 non-null   float64
6   16                  1286 non-null   float64
7   49                  1291 non-null   float64
8   35                  1291 non-null   float64
9   21                  1291 non-null   float64
10  30.47               1290 non-null   float64
11  30.33               1290 non-null   float64
12  30.17               1290 non-null   float64
13  10                  1281 non-null   float64
14  10.1                1281 non-null   float64
15  10.2                1281 non-null   float64
16  14                  1291 non-null   float64
17  4                   1291 non-null   float64
18  21.1                1289 non-null   float64
19  0                   1293 non-null   object
20  0                   1293 non-null   object
dtypes: float64(15), int64(3), object(3)
memory usage: 212.3+ KB

```

VI.- EJERCICIO:

6.1. ejecutar el ítem 2.2, para ello debe crear un archivo 'pandasExcel.xlsx'

Python Tutorial					
PyQt					
Kivy					
Scrapy					
Numpy					
Pandas					

Out[11]:

Python Tutorial	
0	PyQt
1	Kivy
2	Scrapy
3	Numpy
4	Pandas

```
In [13]: type(df)
```

```
Out[13]: pandas.core.frame.DataFrame
```

6.2. ejecutar el ítem 2.3, para ello debe crear un archivo 'Book1.csv'

	A	B	C	D	E	F
1	first name	last name	age			
2	jean	rukebesha	22			
3	susan	anthony	17			
4	jason	james	18			
5						
6						
7						
8						
9						
10						

```
In [14]: pd.read_csv("Book1.csv")
```



```
Out[14]:
```

	first name	last name	age
0	jean	rukebesha	22
1	susan	anthony	17
2	jason	james	18

VII.- BIBLIOGRAFÍA:

- Eugenia Bahit Curso: Python para Principiantes www.eugeniabahit.com
- https://www.youtube.com/watch?v=_8onVOY2j4E

Rubrica:

 UNSA UNIVERSIDAD NACIONAL DE SAN AGUSTÍN DE AREQUIPA	ESCUELA PROFESIONAL DE INGENIERÍA DE TELECOMUNICACIONES LABORATORIO DE INFORMÁTICA EXPERIENCIA: 05 IMPORTAR FICHEROS EXCEL CON PANDAS			Emisión: 28/04/2022 Página 14 / 17	
--	--	--	--	---	--

Puntualidad	Asistencia	Desarrollo de experiencias										Observaciones Conclusiones	Bibliografía	Sugerencias	Total	Porcentaje %
x		5.1	5.2	5.3	5.4	5.5	5.6	5.7	6.1	6.2	6.3					
2.0	2.0	4.0	-	-	-	-	-	-	5.0	5.0	-	-	1.0	1.0	20	15

Ing William Mullisaca
 Docente DAIE






















UNSA
UNIVERSIDAD NACIONAL DE SAN AGUSTÍN DE AREQUIPA

ESCUELA PROFESIONAL DE INGENIERÍA
DE TELECOMUNICACIONES
LABORATORIO DE INFORMÁTICA
EXPERIENCIA: 05
IMPORTAR FICHEROS EXCEL CON
PANDAS

Emisión:
28/04/2022

Página
15 / 17



 UNSA UNIVERSIDAD NACIONAL DE SAN AGUSTÍN DE AREQUIPA		Anexo 1 ATS: Análisis de trabajo seguro					Fecha 30/04/2020			
							Versión 1.1			
							Código ATS20V1			
Curso:		Tarea:				Docente:				
Ambiente:		Grupo:				Mesa:		Fecha:		
Integrantes (Apellidos y nombres)			Firma	Integrantes (Apellidos y nombres)			Firma			
1.				2.						
3.				4.						
Elementos de protección (Marque con aspa)				Características de elementos de protección, equipos y herramientas						
						1. _____				
USO OBLIGATORIO DEL CASCO DE SEGURIDAD	USO OBLIGATORIO DE BOTAS AISLANTES	USO OBLIGATORIO DE GUANTES AISLANTES	USO OBLIGATORIO DE PROTECCIÓN OCULAR	USO OBLIGATORIO DE MASCARILLA	USO OBLIGATORIO DE PROTECTOR FACIAL	2. _____				
						3. _____				
						4. _____				
USO OBLIGATORIO DE PROTECCIÓN AUDITIVA	APAGAR DESCONECTAR CUANDO NO SE USE	USO OBLIGATORIO DE PROTECTOR AJUSTABLE	OBLIGATORIO CONECTAR A TIERRA	USO OBLIGATORIO DE TRAJE DE SEGURIDAD	OTRO	5. _____				
Acciones a realizar (marque con aspa las advertencias por cada acción)										Otros riesgos en las acciones a realizar, considerar medidas de control
1.										1. _____
2.										_____
3.										2. _____
4.										_____
5.										_____



UNSA
UNIVERSIDAD NACIONAL DE SAN AGUSTÍN DE AREQUIPA

ESCUELA PROFESIONAL DE INGENIERÍA
DE TELECOMUNICACIONES
LABORATORIO DE INFORMÁTICA
EXPERIENCIA: 05
IMPORTAR FICHEROS EXCEL CON
PANDAS

Emisión:
28/04/2022

Página
17 / 17



6.

3. _____