IPynbScripts

July 4, 2022

1 Title: Collecting NHS England accident and emergency attendances and admissions data using interactive Jupyter widgets

Author: B210635

2 Load the ‘pandas’ package and test dataset

Load the ‘pandas’ package

[1]:

```
import pandas as pd
```

Import test dataset

[2]:

```
# import the test dataset and store it as in the object 'testData'
testData=pd.read_csv("../Data/ae_attendances_admissions_emergency_test.csv")
# testData
```

Dataset types

Check the data type in the testData data frame using the dtypes function from the Python pandas package.

[3]:

```
result = testData.dtypes #store the data types from the 'testData' frame in the␣
 ↪object 'result'
print("Output:")
print(result) # print the results of the datatypes
# The results showed that the index and admissions variables are stored as␣
 ↪integers while
# the org_code and period variables are stored are strings.
```

```
Output:
index           int64
org_code       object
period         object
admissions      int64
dtype: object
```

Review the first row of the test dataset

```
[4]: testData.head(n=1) # view first row of the dataset using the head() function in␣
      ↪Python
```

```
[4]:    index org_code      period  admissions
     0   2826      RGT  2016-07-01        3123
```

Create an empty data frame in the working data folder to collect the data captured by the Juypter widgets.

```
[5]: # create empty dataframe and store it as the object 'dfTofill'
     dfTofill = pd.DataFrame({'index': [0],# Integer
     'org_code': ['NA'], # String
     'period': [pd.Timestamp('20000101')], # Date
     'admissions': [0], # Integer
     'consent': [False]}) # Boolean
     dfTofill # print the empty dataframe create
```

```
[5]:    index org_code     period  admissions  consent
     0      0       NA 2000-01-01           0    False
```

Save the empty data frame as CSV file to the 'Data' folder and name it 'CollectedData'

```
[6]: #dfTofill.to_csv('../Data/CollectedData.csv', index=False)
```

Read in the empty dataframe to collect the data from the Jupyter-widgets.

```
[7]: CollectData=pd.read_csv("../Data/CollectedData.csv") #store the empty dataframe␣
      ↪as the object 'collected data'
     CollectData #print the new object out
```

```
[7]:    index  org_code      period  admissions  consent
     0      0       NaN  2000-01-01           0    False
```

Review the first row of the test dataset again

```
[8]: testData.head(n=1) #use the head() function in Python to review the first row␣
      ↪of the dataset with n=1
```

```
[8]:    index org_code      period  admissions
     0   2826      RGT  2016-07-01        3123
```

Index variable

The first variable contains the index number, that allows connecting the test data to the orginal data set "../RawData/ae_attendances.csv". Using indexing to add the index number to the 'dfTofill' file.

```
[9]: index_number=2826 #Index number for first record.
     dfTofill.iloc[0,0]=index_number # adding index number to 'dfTofill' file
```

```
dfTofill #print dfTofill file with index number added
```

[9]:

```
   index org_code     period  admissions  consent
0   2826       NA 2000-01-01           0    False
```

3 Widgets

Widgets are graphical user interface element, such as a button, dropdown or textbox. Importing the ipywidgets Python package to use widgets

[10]:

```
#Load the 'ipywidgets' package
import ipywidgets as widgets
```

[11]:

```
#Load the 'IPython.display' package to display different objects in Jupyter␣
 ↪notebook
from IPython.display import display
```

Consent

Here the checkbox widget will be used to obtain consent from end-users to process and share the data collected. The checkbox widget is used to display boolean datatype - True or False

[12]:

```
a = widgets.Checkbox(value=False,
description='I consent for the data provided to be processed and shared',
disabled=False) # use the checkbox widget and store as the object 'a'

display(a) # use the display function to display the widget object
# mark the checkbox to give consent
```

```
Checkbox(value=False, description='I consent for the data provided to be processed and shared'
```

[13]:

```
dfTofill.iloc[0,4]=a.value # Using indexing to add the index number to the␣
 ↪'dfTofill' file for consent variable
dfTofill # print dfTofill file with consent variable equal True
```

[13]:

```
   index org_code     period  admissions  consent
0   2826       NA 2000-01-01           0     True
```

Setup a DatePicker widget to collect period data

[14]:

```
b = widgets.DatePicker(
description='Period',
disabled=False
) # use the DatePicker widget and store as the object 'b'


display(b) # use the display function to display the DatePicker widget object
```

```
#Select "2016-07-01" as first date
```

```
DatePicker(value=None, description='Period')
```

[15]:
```
dfTofill.iloc[0,2]=b.value # Using indexing to add the index number to the␣
 ↪'dfTofill' file for period variable
dfTofill # print dfTofill file with period value added
```

[15]:
```
   index org_code      period  admissions  consent
0   2826       NA  2016-07-01           0     True
```

Use describe the describe() function from the numpy Python package to calculate summary statistics for the testData data frame.

[16]:
```
#Load the 'numpy' package
import numpy as np
# testData.describe(include='all')
```

Applying pandas unique() function to get the unique Organisation data service (ODS) codes in the test data.

[17]:
```
org_code=list(testData['org_code'].unique()) # create a list of unique ODS␣
 ↪codes and store as object 'org_code'
org_code
```

[17]:
```
['RGT', 'RM1', 'RRK', 'RJN', 'RNZ', 'RQ3', 'RBL', 'R1K', 'RPA', 'RA3', 'RDE']
```

Use selection widgets to display single selection lists.

[18]:
```
c=widgets.Select(
    options=org_code,
    value='RGT',
    rows=len(org_code),
    description='ODS code:',
    disabled=False
) # use the Select widget and store as the object 'c'

 # display(c) # use the display function to display the DatePicker widget object

# Select first code
```

[19]:
```
dfTofill.iloc[0,1]=c.value # Using indexing to add the index number to the␣
 ↪'dfTofill' file for 'org_code' variable
dfTofill # print dfTofill file with 'org_code' value added
```

```
[19]:    index org_code      period  admissions  consent
      0   2826      RGT  2016-07-01           0     True
```

Setup numeric widgets using the 'IntText' widget because the admissions data type is an integer.

```
[20]: d=widgets.IntText(
          value=0,
          description='Admissions:',
          disabled=False) # use the integer widget and store as the object 'd'

      display(d) # use the display function to display the integer widget object

      #select 3123 as value for admissions from the first row
```

```
IntText(value=0, description='Admissions:')
```

```
[21]: dfTofill.iloc[0,3]=d.value # Using indexing to add the index number to the␣
       ↪'dfTofill' file for 'admissions' variable
      dfTofill # print dfTofill file with 'admissions' value added
```

```
[21]:    index org_code      period  admissions  consent
      0   2826      RGT  2016-07-01        3123     True
```

4 Concatenating the collected data to the CollectData data frame.

Use of the concat() function from the Python pandas package to append the CollectData and dfTofill data frames.

```
[22]: # CollectData is the first data frame
      # dfTofill is the second data frame
      CollectData = pd.concat([CollectData, dfTofill]) # concatenate the collected␣
       ↪data to the CollectData data frame
      # display(CollectData) # display results
```

Ensuring consent before saving the collected data to the working data folder.

```
[23]: CollectData=CollectData[CollectData['consent'] == True] # subset the␣
       ↪concatenated data to ensure consent
      display(CollectData) # display results
```

```
   index org_code      period  admissions  consent
0   2826      RGT  2016-07-01        3123     True
```

Saving the data collected by the data capture tool to the working data folder.

[24]:
```
CollectData.to_csv('../Data/CollectedData.csv', index=False)
```

5 Iterate data collection process for second row of data

Review the second row of the test dataset

[25]:
```
# testData.head(n=2) #use the head() function in Python to review the first row␣
 ↪of the dataset with n=2
```

Index variable for second row

[26]:
```
index_number=2865 #Index number for second record.
dfTofill.iloc[0,0]=index_number # adding index number to 'dfTofill' file
dfTofill #print dfTofill2 file with index number added
```

[26]:
```
   index org_code      period  admissions  consent
0   2865      RGT  2016-07-01        3123     True
```

Consent using the checkbox widget to display boolean datatype - True or False

[27]:
```
a1 = widgets.Checkbox(value=False,
description='I consent for the data provided to be processed and shared',
disabled=False) # use the checkbox widget and store as the object 'a1'

display(a1) # use the display function to display the widget object
# mark the checkbox to give consent
```

```
Checkbox(value=False, description='I consent for the data provided to be processed and shared'
```

[28]:
```
dfTofill.iloc[0,4]=a1.value # Using indexing to add the index number to the␣
 ↪'dfTofill' file for consent variable
dfTofill # print dfTofill file with consent variable equal True
```

[28]:
```
   index org_code      period  admissions  consent
0   2865      RGT  2016-07-01        3123     True
```

Setup a DatePicker widget to collect period data for second row

[29]:
```
b1 = widgets.DatePicker(
description='Period',
disabled=False
) # use the DatePicker widget and store as the object 'b1'


display(b1) # use the display function to display the DatePicker widget object

#Select "2016-07-01" as date for second row
```

```
DatePicker(value=None, description='Period')
```

[30]:

```
dfTofill.iloc[0,2]=b1.value # Using indexing to add the index number to the␣
 ↪'dfTofill' file for period variable
dfTofill # print dfTofill file with period value added
```

[30]:

```
   index org_code      period  admissions  consent
0   2865      RGT  2016-07-01        3123     True
```

Use selection widgets to display single selection lists for row 2 for org_code variable.

[31]:

```
c1=widgets.Select(
    options=org_code,
    value='RM1',
    rows=len(org_code),
    description='ODS code:',
    disabled=False
) # use the Select widget and store as the object 'c1'

# display(c1) # use the display function to display the DatePicker widget object

# Select second code
```

[32]:

```
dfTofill.iloc[0,1]=c1.value # Using indexing to add the index number to row2␣
 ↪'dfTofill' file for 'org_code' variable
dfTofill # print dfTofill file row2 with 'org_code' value added
```

[32]:

```
   index org_code      period  admissions  consent
0   2865      RM1  2016-07-01        3123     True
```

Use IntText' widget to collect index value for admissions for second row.

[33]:

```
d1=widgets.IntText(
    value=0,
    description='Admissions:',
    disabled=False) # use the integer widget and store as the object 'd1'

display(d1) # use the display function to display the integer widget object

#select 2891 as value for admissions from the second row
```

```
IntText(value=0, description='Admissions:')
```

[34]:

```
dfTofill.iloc[0,3]=d1.value # Using indexing to add the index number to the␣
 ↪'dfTofill' file for 'admissions' variable
dfTofill # print dfTofill file with 'admissions' value added
```

[34]:

```
   index org_code      period  admissions  consent
0   2865      RM1  2016-07-01        2891     True
```

Concatenate second row of data to the CollectData data frame.

[35]:

```python
CollectData = pd.concat([CollectData, dfTofill]) # concatenate the collected data to the CollectData data frame
# display(CollectData) # display results
```

6 Iterate data collection process for third row of data

Review the third row of the test dataset

[36]:

```python
# testData.head(n=3) #use the head() function in Python to review the third row of the dataset with n=3
```

Index variable for third row

[37]:

```python
index_number=3941 #Index number for third record.
dfTofill.iloc[0,0]=index_number # adding index number to 'dfTofill' file
dfTofill #print dfTofill file with index number added
```

[37]:

```
   index org_code      period  admissions  consent
0   3941      RM1  2016-07-01        2891     True
```

Consent using the checkbox widget to display boolean datatype - True or False

[38]:

```python
a2 = widgets.Checkbox(value=False,
description='I consent for the data provided to be processed and shared',
disabled=False) # use the checkbox widget and store as the object 'a2'

display(a2) # use the display function to display the widget object
# mark the checkbox to give consent
```

```
Checkbox(value=False, description='I consent for the data provided to be processed and shared'
```

[39]:

```python
dfTofill.iloc[0,4]=a2.value # Using indexing to add the index number to the 'dfTofill' file for consent variable
dfTofill # print dfTofill file with consent variable equal True
```

[39]:

```
   index org_code      period  admissions  consent
0   3941      RM1  2016-07-01        2891     True
```

Setup a DatePicker widget to collect period data for third row

[40]:

```python
b2 = widgets.DatePicker(
description='Period',
disabled=False
) # use the DatePicker widget and store as the object 'b1'

display(b2) # use the display function to display the DatePicker widget object
#Select "2016-04-01" as date for third row
```

```
DatePicker(value=None, description='Period')
```

[41]:

```
dfTofill.iloc[0,2]=b2.value # Using indexing to add the index number to the␣
 ↪'dfTofill' file for period variable
dfTofill # print dfTofill file with period value added
```

[41]:

```
   index org_code      period  admissions  consent
0   3941      RM1  2016-04-01        2891     True
```

Use selection widgets to display single selection lists for row 3 for org_code variable.

[42]:

```
c2=widgets.Select(
    options=org_code,
    value='RRK',
    rows=len(org_code),
    description='ODS code:',
    disabled=False
) # use the Select widget and store as the object 'c1'

# display(c2) # use the display function to display the DatePicker widget object

# Select third code
```

[43]:

```
dfTofill.iloc[0,1]=c2.value # Using indexing to add the index number to row3␣
 ↪'dfTofill' file for 'org_code' variable
dfTofill # print dfTofill file row3 with 'org_code' value added
```

[43]:

```
   index org_code      period  admissions  consent
0   3941      RRK  2016-04-01        2891     True
```

Use IntText' widget to collect index value for admissions for third row.

[44]:

```
d2=widgets.IntText(
    value=0,
    description='Admissions:',
    disabled=False) # use the integer widget and store as the object 'd1'

display(d2) # use the display function to display the integer widget object

#select 2744 as value for admissions from the third row
```

```
IntText(value=0, description='Admissions:')
```

[45]:
```
dfTofill.iloc[0,3]=d2.value # Using indexing to add the index number to the␣
 ↪'dfTofill' file for 'admissions' variable
dfTofill # print dfTofill file with 'admissions' value added
```

[45]:
```
   index org_code      period  admissions  consent
0   3941      RRK  2016-04-01        2744     True
```

Concatenate third row of data to the CollectData data frame.

[46]:
```
CollectData = pd.concat([CollectData, dfTofill]) # concatenate the collected␣
 ↪data to the CollectData data frame
# display(CollectData) # display results
```

7 Iterate data collection process for fourth row of data

[47]:
```
# testData.head(n=4) #use the head() function in Python to review the fourth␣
 ↪row of the dataset with n=4
```

[48]:
```
index_number=5431 #Index number for fourth record.
dfTofill.iloc[0,0]=index_number # adding index number to 'dfTofill' file
dfTofill #print dfTofill file with index number added
```

[48]:
```
   index org_code      period  admissions  consent
0   5431      RRK  2016-04-01        2744     True
```

[49]:
```
#setup consent using checkbox widget for the fourth row
a3 = widgets.Checkbox(value=False,
description='I consent for the data provided to be processed and shared',
disabled=False) # use the checkbox widget and store as the object 'a3'

display(a3) # use the display function to display the widget object
# mark the checkbox to give consent
```

```
Checkbox(value=False, description='I consent for the data provided to be processed and shared'
```

[50]:
```
dfTofill.iloc[0,4]=a2.value # Using indexing to add the index number to the␣
 ↪'dfTofill' file for consent variable
dfTofill # print dfTofill file with consent variable equal True
```

[50]:
```
   index org_code      period  admissions  consent
0   5431      RRK  2016-04-01        2744     True
```

[51]:
```
#Setup a DatePicker widget to collect period data for the fourth row
```

```
b3 = widgets.DatePicker(
description='Period',
disabled=False
) # use the DatePicker widget and store as the object 'b3'

display(b3) # use the display function to display the DatePicker widget object

#Select "2017-12-01" as date for fourth row
```

```
DatePicker(value=None, description='Period')
```

[52]:

```
dfTofill.iloc[0,2]=b3.value # Using indexing to add the index number to the␣
 ↪'dfTofill' file for period variable
dfTofill # print dfTofill file with period value added
```

[52]:

```
   index org_code      period  admissions  consent
0   5431      RRK  2017-12-01        2744     True
```

[53]:

```
#Use selection widgets to display single selection lists for row 4 for org_code␣
 ↪variable.
c3=widgets.Select(
    options=org_code,
    value='RJN',
    rows=len(org_code),
    description='ODS code:',
    disabled=False
) # use the Select widget and store as the object 'c3'

# display(c3) # use the display function to display the DatePicker widget object

# Select fourth code
```

[54]:

```
dfTofill.iloc[0,1]=c3.value # Using indexing to add the index number to row4␣
 ↪'dfTofill' file for 'org_code' variable
dfTofill # print dfTofill file row4 with 'org_code' value added
```

[54]:

```
   index org_code      period  admissions  consent
0   5431      RJN  2017-12-01        2744     True
```

[55]:

```
#Use IntText' widget to collect index value for admissions for fourth row.

d3=widgets.IntText(
    value=0,
    description='Admissions:',
    disabled=False) # use the integer widget and store as the object 'd3'
```

```
display(d3) # use the display function to display the integer widget object

#select 971 as value for admissions from the fourth row
```

```
IntText(value=0, description='Admissions:')
```

[56]:
```
dfTofill.iloc[0,3]=d3.value # Using indexing to add the index number to the␣
 ↪'dfTofill' file for 'admissions' variable
dfTofill # print dfTofill file with 'admissions' value added

#Concatenate fourth row of data to the CollectData data frame.
CollectData = pd.concat([CollectData, dfTofill]) # concatenate the collected␣
 ↪data to the CollectData data frame
# display(CollectData) # display results
```

8 Iterate data collection process for fifth row of data

[57]:
```
 # testData.head(n=5) #use the head() function in Python to review the fifth␣
 ↪row of the dataset with n=5
```

[58]:
```
index_number=5940 #Index number for fifth record.
dfTofill.iloc[0,0]=index_number # adding index number to 'dfTofill' file
dfTofill #print dfTofill file with index number added
```

[58]:
```
   index org_code      period  admissions  consent
0   5940      RJN  2017-12-01         971     True
```

[59]:
```
#setup consent using checkbox widget for the fifth row
a4 = widgets.Checkbox(value=False,
description='I consent for the data provided to be processed and shared',
disabled=False) # use the checkbox widget and store as the object 'a4'

display(a4) # use the display function to display the widget object
# mark the checkbox to give consent
```

```
Checkbox(value=False, description='I consent for the data provided to be processed and shared'
```

[60]:
```
dfTofill.iloc[0,4]=a4.value # Using indexing to add the index number to the␣
 ↪'dfTofill' file for consent variable
dfTofill # print dfTofill file with consent variable equal True

#Setup a DatePicker widget to collect period data for the fifth row

b4 = widgets.DatePicker(
description='Period',
disabled=False
) # use the DatePicker widget and store as the object 'b4'


display(b4) # use the display function to display the DatePicker widget object

#Select "2017-11-01" as date for fifth row
```

```
DatePicker(value=None, description='Period')
```

[61]:

```
dfTofill.iloc[0,2]=b4.value # Using indexing to add the index number to the␣
 ↪'dfTofill' file for period variable
dfTofill # print dfTofill file with period value added


#Use selection widgets to display single selection lists for row 5 for org_code␣
 ↪variable.
c4=widgets.Select(
    options=org_code,
    value='RNZ',
    rows=len(org_code),
    description='ODS code:',
    disabled=False
) # use the Select widget and store as the object 'c4'

# display(c4) # use the display function to display the DatePicker widget object

# Select fifith code
```

[62]:

```
dfTofill.iloc[0,1]=c4.value # Using indexing to add the index number to row5␣
 ↪'dfTofill' file for 'org_code' variable
dfTofill # print dfTofill file row5 with 'org_code' value added

#Use IntText' widget to collect index value for admissions for fourth row.

d4=widgets.IntText(
    value=0,
    description='Admissions:',
    disabled=False) # use the integer widget and store as the object 'd4'

display(d4) # use the display function to display the integer widget object

#select 1131 as value for admissions from the fifth row
```

```
IntText(value=0, description='Admissions:')
```

```
[63]: dfTofill.iloc[0,3]=d4.value # Using indexing to add the index number to the␣
       ↪'dfTofill' file for 'admissions' variable
      dfTofill # print dfTofill file with 'admissions' value added

      #Concatenate fifth row of data to the CollectData data frame.
      CollectData = pd.concat([CollectData, dfTofill]) # concatenate the collected␣
       ↪data to the CollectData data frame
      # display(CollectData) # display results
```

9 Iterate data collection process for sixth row of data

```
[64]: # testData.head(n=6) #use the head() function in Python to review the sixth row␣
       ↪of the dataset with n=6
```

```
[65]: index_number=6727 #Index number for sixth record.
      dfTofill.iloc[0,0]=index_number # adding index number to 'dfTofill' file
      dfTofill #print dfTofill file with index number added

      #setup consent using checkbox widget for the sixth row
      a5 = widgets.Checkbox(value=False,
      description='I consent for the data provided to be processed and shared',
      disabled=False) # use the checkbox widget and store as the object 'a5'

      display(a5) # use the display function to display the widget object
      # mark the checkbox to give consent
```

```
Checkbox(value=False, description='I consent for the data provided to be processed and shared')
```

```
[66]: dfTofill.iloc[0,4]=a5.value # Using indexing to add the index number to the␣
       ↪'dfTofill' file for consent variable
      dfTofill # print dfTofill file with consent variable equal True

      #Setup a DatePicker widget to collect period data for the sixth row

      b5 = widgets.DatePicker(
      description='Period',
      disabled=False
      ) # use the DatePicker widget and store as the object 'b5'

      display(b5) # use the display function to display the DatePicker widget object

      #Select "2017-08-01" as date for sixth row
```

```
DatePicker(value=None, description='Period')
```

[67]:

```python
dfTofill.iloc[0,2]=b5.value # Using indexing to add the index number to the 'dfTofill' file for period variable
dfTofill # print dfTofill file with period value added


#Use selection widgets to display single selection lists for row 6 for org_code variable.
c5=widgets.Select(
    options=org_code,
    value='RQ3',
    rows=len(org_code),
    description='ODS code:',
    disabled=False
) # use the Select widget and store as the object 'c5'

# display(c5) # use the display function to display the DatePicker widget object

# Select sixth code
```

[68]:

```python
dfTofill.iloc[0,1]=c5.value # Using indexing to add the index number to row6 'dfTofill' file for 'org_code' variable
dfTofill # print dfTofill file row6 with 'org_code' value added

#Use IntText' widget to collect index value for admissions for sixth row.

d5=widgets.IntText(
    value=0,
    description='Admissions:',
    disabled=False) # use the integer widget and store as the object 'd5'

display(d5) # use the display function to display the integer widget object

#select 754 as value for admissions from the sixth row
```

```
IntText(value=0, description='Admissions:')
```

[69]:

```python
dfTofill.iloc[0,3]=d5.value # Using indexing to add the index number to the 'dfTofill' file for 'admissions' variable
dfTofill # print dfTofill file with 'admissions' value added

#Concatenate sixth row of data to the CollectData data frame.
CollectData = pd.concat([CollectData, dfTofill]) # concatenate the collected data to the CollectData data frame
# display(CollectData) # display results
```

10 Iterate data collection process for seventh row of data

[70]:
```
# testData.head(n=7) #use the head() function in Python to review the seventh␣
 ↪row of the dataset with n=7
```

[71]:
```
index_number=7952 #Index number for seventh record.
dfTofill.iloc[0,0]=index_number # adding index number to 'dfTofill' file
dfTofill #print dfTofill file with index number added

#setup consent using checkbox widget for the seventh row
a6 = widgets.Checkbox(value=False,
description='I consent for the data provided to be processed and shared',
disabled=False) # use the checkbox widget and store as the object 'a6'

display(a6) # use the display function to display the widget object
# mark the checkbox to give consent
```

```
Checkbox(value=False, description='I consent for the data provided to be processed and shared')
```

[72]:
```
dfTofill.iloc[0,4]=a6.value # Using indexing to add the index number to the␣
 ↪'dfTofill' file for consent variable
dfTofill # print dfTofill file with consent variable equal True

#Setup a DatePicker widget to collect period data for the seventh row

b6 = widgets.DatePicker(
description='Period',
disabled=False
) # use the DatePicker widget and store as the object 'b6'


display(b6) # use the display function to display the DatePicker widget object

#Select "2017-05-01" as date for seventh row
```

```
DatePicker(value=None, description='Period')
```

[73]:
```
dfTofill.iloc[0,2]=b6.value # Using indexing to add the index number to the␣
 ↪'dfTofill' file for period variable
dfTofill # print dfTofill file with period value added


#Use selection widgets to display single selection lists for row 7 for org_code␣
 ↪variable.
c6=widgets.Select(
    options=org_code,
    value='RBL',
    rows=len(org_code),
    description='ODS code:',
    disabled=False
) # use the Select widget and store as the object 'c6'

# display(c6) # use the display function to display the DatePicker widget object

# Select seventh code
```

[74]:

```
dfTofill.iloc[0,1]=c6.value # Using indexing to add the index number to row7␣
 ↪'dfTofill' file for 'org_code' variable
dfTofill # print dfTofill file row7 with 'org_code' value added

#Use IntText' widget to collect index value for admissions for seventh row.

d6=widgets.IntText(
    value=0,
    description='Admissions:',
    disabled=False) # use the integer widget and store as the object 'd6'

display(d6) # use the display function to display the integer widget object

#select 2491 as value for admissions from the seventh row
```

```
IntText(value=0, description='Admissions:')
```

[75]:

```
dfTofill.iloc[0,3]=d6.value # Using indexing to add the index number to the␣
 ↪'dfTofill' file for 'admissions' variable
dfTofill # print dfTofill file with 'admissions' value added

#Concatenate seventh row of data to the CollectData data frame.
CollectData = pd.concat([CollectData, dfTofill]) # concatenate the collected␣
 ↪data to the CollectData data frame
# display(CollectData) # display results
```

11 Iterate data collection process for eighth row of data

[76]:

```
# testData.head(n=8) #use the head() function in Python to review the eighth␣
 ↪row of the dataset with n=8
```

[77]:

```
index_number=8427 #Index number for eighth record.
dfTofill.iloc[0,0]=index_number # adding index number to 'dfTofill' file
dfTofill #print dfTofill file with index number added
```

```
#setup consent using checkbox widget for the eighth row
a7 = widgets.Checkbox(value=False,
description='I consent for the data provided to be processed and shared',
disabled=False) # use the checkbox widget and store as the object 'a7'

display(a7) # use the display function to display the widget object
# mark the checkbox to give consent
```

```
Checkbox(value=False, description='I consent for the data provided to be processed and shared')
```

[78]:

```
dfTofill.iloc[0,4]=a6.value # Using indexing to add the index number to the␣
 ↪'dfTofill' file for consent variable
dfTofill # print dfTofill file with consent variable equal True

#Setup a DatePicker widget to collect period data for the eighth row

b7 = widgets.DatePicker(
description='Period',
disabled=False
) # use the DatePicker widget and store as the object 'b7'


display(b7) # use the display function to display the DatePicker widget object

#Select "2019-03-01" as date for eighth row
```

```
DatePicker(value=None, description='Period')
```

[79]:

```
dfTofill.iloc[0,2]=b7.value # Using indexing to add the index number to the␣
 ↪'dfTofill' file for period variable
dfTofill # print dfTofill file with period value added


#Use selection widgets to display single selection lists for row 8 for org_code␣
 ↪variable.
c7=widgets.Select(
    options=org_code,
    value='R1K',
    rows=len(org_code),
    description='ODS code:',
    disabled=False
) # use the Select widget and store as the object 'c7'

# display(c7) # use the display function to display the DatePicker widget object

# Select eighth code
```

[80]:

```
dfTofill.iloc[0,1]=c7.value # Using indexing to add the index number to row8␣
 ↪'dfTofill' file for 'org_code' variable
dfTofill # print dfTofill file row8 with 'org_code' value added

#Use IntText' widget to collect index value for admissions for eighth row.

d7=widgets.IntText(
    value=0,
    description='Admissions:',
    disabled=False) # use the integer widget and store as the object 'd7'

display(d7) # use the display function to display the integer widget object

#select 6522 as value for admissions from the eighth row
```

```
IntText(value=0, description='Admissions:')
```

[81]:

```
dfTofill.iloc[0,3]=d7.value # Using indexing to add the index number to the␣
 ↪'dfTofill' file for 'admissions' variable
dfTofill # print dfTofill file with 'admissions' value added

#Concatenate eighth row of data to the CollectData data frame.
CollectData = pd.concat([CollectData, dfTofill]) # concatenate the collected␣
 ↪data to the CollectData data frame
# display(CollectData) # display results
```

12 Iterate data collection process for ninth row of data

[82]:

```
# testData.head(n=9) #use the head() function in Python to review the ninth row␣
 ↪of the dataset with n=9
```

[83]:

```
index_number=10489 #Index number for ninth record.
dfTofill.iloc[0,0]=index_number # adding index number to 'dfTofill' file
dfTofill #print dfTofill file with index number added

#setup consent using checkbox widget for the eighth row
a8 = widgets.Checkbox(value=False,
description='I consent for the data provided to be processed and shared',
disabled=False) # use the checkbox widget and store as the object 'a8'

display(a8) # use the display function to display the widget object
# mark the checkbox to give consent
```

```
Checkbox(value=False, description='I consent for the data provided to be processed and shared'
```

[84]:

```
dfTofill.iloc[0,4]=a8.value # Using indexing to add the index number to the␣
 ↪'dfTofill' file for consent variable
dfTofill # print dfTofill file with consent variable equal True

#Setup a DatePicker widget to collect period data for the ninth row

b8 = widgets.DatePicker(
description='Period',
disabled=False
) # use the DatePicker widget and store as the object 'b8'


display(b8) # use the display function to display the DatePicker widget object

#Select "2018-10-01" as date for ninth row
```

```
DatePicker(value=None, description='Period')
```

[85]:

```
dfTofill.iloc[0,2]=b8.value # Using indexing to add the index number to the␣
 ↪'dfTofill' file for period variable
dfTofill # print dfTofill file with period value added


#Use selection widgets to display single selection lists for row 9 for org_code␣
 ↪variable.
c8=widgets.Select(
    options=org_code,
    value='RPA',
    rows=len(org_code),
    description='ODS code:',
    disabled=False
) # use the Select widget and store as the object 'c8'

# display(c8) # use the display function to display the DatePicker widget object

# Select ninth code
```

[86]:

```
dfTofill.iloc[0,1]=c8.value # Using indexing to add the index number to row9␣
 ↪'dfTofill' file for 'org_code' variable
dfTofill # print dfTofill file row9 with 'org_code' value added

#Use IntText' widget to collect index value for admissions for ninth row.

d8=widgets.IntText(
    value=0,
    description='Admissions:',
    disabled=False) # use the integer widget and store as the object 'd8'

display(d8) # use the display function to display the integer widget object

#select 2065 as value for admissions from the ninth row
```

```
IntText(value=0, description='Admissions:')
```

[87]:

```
dfTofill.iloc[0,3]=d8.value # Using indexing to add the index number to the␣
 ↪'dfTofill' file for 'admissions' variable
dfTofill # print dfTofill file with 'admissions' value added

#Concatenate ninth row of data to the CollectData data frame.
CollectData = pd.concat([CollectData, dfTofill]) # concatenate the collected␣
 ↪data to the CollectData data frame
# display(CollectData) # display results
```

13 Iterate data collection process for tenth row of data

[88]:

```
# testData.head(n=10) #use the head() function in Python to review the tenth␣
 ↪row of the dataset with n=10
```

[89]:

```
index_number=12390 #Index number for tenth record.
dfTofill.iloc[0,0]=index_number # adding index number to 'dfTofill' file
dfTofill #print dfTofill file with index number added

#setup consent using checkbox widget for the tenth row
a9 = widgets.Checkbox(value=False,
description='I consent for the data provided to be processed and shared',
disabled=False) # use the checkbox widget and store as the object 'a9'

display(a9) # use the display function to display the widget object
# mark the checkbox to give consent
```

```
Checkbox(value=False, description='I consent for the data provided to be processed and shared')
```

[90]:

```
dfTofill.iloc[0,4]=a9.value # Using indexing to add the index number to the␣
 ↪'dfTofill' file for consent variable
dfTofill # print dfTofill file with consent variable equal True

#Setup a DatePicker widget to collect period data for the tenth row

b9 = widgets.DatePicker(
description='Period',
disabled=False
) # use the DatePicker widget and store as the object 'b9'


display(b9) # use the display function to display the DatePicker widget object

#Select "2018-05-01" as date for tenth row
```

```
DatePicker(value=None, description='Period')
```

[91]:

```
dfTofill.iloc[0,2]=b9.value # Using indexing to add the index number to the␣
 ↪'dfTofill' file for period variable
dfTofill # print dfTofill file with period value added


#Use selection widgets to display single selection lists for row 10 for␣
 ↪org_code variable.
c9=widgets.Select(
    options=org_code,
    value='RA3',
    rows=len(org_code),
    description='ODS code:',
    disabled=False
) # use the Select widget and store as the object 'c9'

# display(c9) # use the display function to display the DatePicker widget object

# Select tenth code
```

[92]:

```
dfTofill.iloc[0,1]=c9.value # Using indexing to add the index number to row10␣
 ↪'dfTofill' file for 'org_code' variable
dfTofill # print dfTofill file row10 with 'org_code' value added

#Use IntText' widget to collect index value for admissions for tenth row.

d9=widgets.IntText(
    value=0,
    description='Admissions:',
    disabled=False) # use the integer widget and store as the object 'd9'

display(d9) # use the display function to display the integer widget object

#select 1090 as value for admissions from the tenth row
```

```
IntText(value=0, description='Admissions:')
```

[93]:
```
dfTofill.iloc[0,3]=d9.value # Using indexing to add the index number to the␣
 ↪'dfTofill' file for 'admissions' variable
dfTofill # print dfTofill file with 'admissions' value added

#Concatenate tenth row of data to the CollectData data frame.
CollectData = pd.concat([CollectData, dfTofill]) # concatenate the collected␣
 ↪data to the CollectData data frame
# display(CollectData) # display results
```

14 Iterate data collection process for eleventh row of data

[94]:
```
# testData.head(n=11) #use the head() function in Python to review the eleventh␣
 ↪row of the dataset with n=11
```

[95]:
```
index_number=12463 #Index number for eleventh record.
dfTofill.iloc[0,0]=index_number # adding index number to 'dfTofill' file
dfTofill #print dfTofill file with index number added

#setup consent using checkbox widget for the eleventh row
a10 = widgets.Checkbox(value=False,
description='I consent for the data provided to be processed and shared',
disabled=False) # use the checkbox widget and store as the object 'a10'

display(a10) # use the display function to display the widget object
# mark the checkbox to give consent
```

```
Checkbox(value=False, description='I consent for the data provided to be processed and shared')
```

[96]:
```
dfTofill.iloc[0,4]=a10.value # Using indexing to add the index number to the␣
 ↪'dfTofill' file for consent variable
dfTofill # print dfTofill file with consent variable equal True

#Setup a DatePicker widget to collect period data for the eleventh row

b10 = widgets.DatePicker(
description='Period',
disabled=False
) # use the DatePicker widget and store as the object 'b10'


display(b10) # use the display function to display the DatePicker widget object

#Select "2018-04-01" as date for eleventh row
```

```
DatePicker(value=None, description='Period')
```

[97]:

```
dfTofill.iloc[0,2]=b10.value # Using indexing to add the index number to the␣
 ↪'dfTofill' file for period variable
dfTofill # print dfTofill file with period value added


#Use selection widgets to display single selection lists for row 11 for␣
 ↪org_code variable.
c10=widgets.Select(
    options=org_code,
    value='RDE',
    rows=len(org_code),
    description='ODS code:',
    disabled=False
) # use the Select widget and store as the object 'c10'

# display(c10) # use the display function to display the DatePicker widget␣
 ↪object

# Select eleventh code
```

[98]:

```
dfTofill.iloc[0,1]=c10.value # Using indexing to add the index number to row11␣
 ↪'dfTofill' file for 'org_code' variable
dfTofill # print dfTofill file row11 with 'org_code' value added

#Use IntText' widget to collect index value for admissions for eleventh row.

d10=widgets.IntText(
    value=0,
    description='Admissions:',
    disabled=False) # use the integer widget and store as the object 'd10'

display(d10) # use the display function to display the integer widget object

#select 2398 as value for admissions from the eleventh row
```

```
IntText(value=0, description='Admissions:')
```

[99]:

```
dfTofill.iloc[0,3]=d10.value # Using indexing to add the index number to the␣
 ↪'dfTofill' file for 'admissions' variable
dfTofill # print dfTofill file with 'admissions' value added

#Concatenate eleventh row of data to the CollectData data frame.
CollectData = pd.concat([CollectData, dfTofill]) # concatenate the collected␣
 ↪data to the CollectData data frame
display(CollectData) # display results
```

```
   index org_code      period  admissions  consent
0   2826      RGT  2016-07-01        3123     True
0   2865      RM1  2016-07-01        2891     True
0   3941      RRK  2016-04-01        2744     True
0   5431      RJN  2017-12-01         971     True
0   5940      RNZ  2017-11-01        1131     True
0   6727      RQ3  2017-08-01         754     True
0   7952      RBL  2017-05-01        2491     True
0   8427      R1K  2019-03-01        6522     True
0  10489      RPA  2018-10-10        2065     True
0  12390      RA3  2018-05-01        1090     True
0  12463      RDE  2018-04-01        2398     True
```

[100]:

```
# Saving the CollectData data frame to the working data folder
CollectData.to_csv('../Data/CollectedData.csv', index=False)
```

[101]:

```
# Saving the captured test data to the 'RawData' folder.

CollectData.to_csv('../RawData/CollectedDataFinal.csv', index=False)
```