Objective: This worksheet shows few different ways to create Pandas DataFrames. **All data shown in this worksheet is fictitious.**

- Let us first create a DataFrame object from two Series objects. The two series objects are shown in the table below:

	Karnataka	Tamil Nadu	Andhra Pradesh	Telangana	Kerala
Capital	Bangalore	Chennai	Amravati	Hyderabad	Thiruvananthapuram
Population	191791	130058	162975	112077	38863

- The corresponding code for creating the Series objects is also shown below:

```
capital = pd.Series (data = ['Bangalore', 'Chennai', 'Amravati', 'Hyderabad','Thiruvananthapuram'],
                     index = ['Karnataka', 'Tamil Nadu', 'Andhra Pradesh', 'Telangana', 'Kerala'])
population = pd.Series(data = [191791, 130058, 162975, 112077, 38863],
                       index = ['Karnataka', 'Tamil Nadu', 'Andhra Pradesh', 'Telangana', 'Kerala'])
```

- The requirement is how to create a DataFrame object from the above two Series. It can be done using the Dictionary data structure as an input parameter to the ***DataFrame()*** function.

```
states = pd.DataFrame(
    {'Capital':capital,
     'Population':population
    }
)
states
```

	Capital	Population
Karnataka	Bangalore	191791
Tamil Nadu	Chennai	130058
Andhra Pradesh	Amravati	162975
Telangana	Hyderabad	112077
Kerala	Thiruvananthapuram	38863

- Notice that the values in the dictionary are the individual Series objects created before. The key names can be programmer decided.
- Now the different elements can be accessed from this DataFrame using ***loc*** and ***iloc***.

```
states.loc['Karnataka','Capital']
```

```
'Bangalore'
```

```
states.iloc[1:3,0]
```

```
Tamil Nadu          Chennai
Andhra Pradesh     Amravati
Name: Capital, dtype: object
```

- Let us say there is another series (3rd) that tells the current chief ministers of these states:

```
cm = pd.Series(data = ['BSY', 'EKGP', 'YSR', 'KCR', 'PV'],
               index = ['Karnataka', 'Tamil Nadu', 'Andhra Pradesh', 'Telangana', 'Kerala'])
cm
```

```
Karnataka           BSY
Tamil Nadu         EKGP
Andhra Pradesh      YSR
Telangana           KCR
Kerala               PV
dtype: object
```

- How this can be added as the 3rd column to the states DataFrame? The procedure is simple as shown below:

```
states['CM']=cm
```

```
states
```

	Capital	Population	CM
Karnataka	Bangalore	191791	BSY
Tamil Nadu	Chennai	130058	EKGP
Andhra Pradesh	Amravati	162975	YSR
Telangana	Hyderabad	112077	KCR
Kerala	Thiruvananthapuram	38863	PV

- Let us say, there is a need to add one more column on the count of IT parks in each state. This data is not yet available but it needs to be created with an initialized value as 0 for each state. The objective can be achieved using a scalar broadcasting:

```
states['IT Parks'] = 0
states
```

	Capital	Population	CM	IT Parks
Karnataka	Bangalore	191791	BSY	0
Tamil Nadu	Chennai	130058	EKGP	0
Andhra Pradesh	Amravati	162975	YSR	0
Telangana	Hyderabad	112077	KCR	0
Kerala	Thiruvananthapuram	38863	PV	0

- Later values can be assigned whenever they are available. For example:

```
states.loc['Karnataka','IT Parks'] = 24
states.loc['Telangana','IT Parks'] = 11
states
```

	Capital	Population	CM	IT Parks
Karnataka	Bangalore	191791	BSY	24
Tamil Nadu	Chennai	130058	EKGP	0
Andhra Pradesh	Amravati	162975	YSR	0
Telangana	Hyderabad	112077	KCR	11
Kerala	Thiruvananthapuram	38863	PV	0

- Individual Series from the created DataFrame can be extracted as shown below:

```
print(states['Capital'])
print(type(states['Capital']))
```

```
Karnataka                    Bangalore
Tamil Nadu                     Chennai
Andhra Pradesh                Amravati
Telangana                     Hyderabad
Kerala             Thiruvananthapuram
Name: Capital, dtype: object
<class 'pandas.core.series.Series'>
```

- A new row can also be appended. Let us say, there is a new row for Maharashtra state. First a Series for Maharashtra is to be created. Notice that a new argument ***Name*** is also provided while creating the Series.

```
maha = pd.Series(data = {'Capital':'Mumbai', 'Population':114200, 'CM':'UT', 'IT Parks':10}, name = 'Maharashtra')
maha
```

```
Capital       Mumbai
Population    114200
CM                UT
IT Parks          10
Name: Maharashtra, dtype: object
```

- Now this series can be appended using the ***append ()*** function.

```
states = states.append(maha)
states
```

	Capital	Population	CM	IT Parks
Karnataka	Bangalore	191791	BSY	24
Tamil Nadu	Chennai	130058	EKGP	0
Andhra Pradesh	Amravati	162975	YSR	0
Telangana	Hyderabad	112077	KCR	11
Kerala	Thiruvananthapuram	38863	PV	0
Maharashtra	Mumbai	114200	UT	10

- A column can be dropped using the ***drop ()*** function. Rows are considered as axis 0 and columns are considered for axis 1. This is also provided as an argument:

```
states = states.drop('IT Parks', axis=1)
```

```
states
```

	Capital	Population	CM
Karnataka	Bangalore	191791	BSY
Tamil Nadu	Chennai	130058	EKGP
Andhra Pradesh	Amravati	162975	YSR
Telangana	Hyderabad	112077	KCR
Kerala	Thiruvananthapuram	38863	PV
Maharashtra	Mumbai	114200	UT

- A row can also be dropped using the ***drop ()*** function. The argument axis is not required because Maharashtra identifies a unique row. However it can also be provided as axis = 0 to the function.

```
states = states.drop('Maharashtra')
states
```

	Capital	Population	CM
Karnataka	Bangalore	191791	BSY
Tamil Nadu	Chennai	130058	EKGP
Andhra Pradesh	Amravati	162975	YSR
Telangana	Hyderabad	112077	KCR
Kerala	Thiruvananthapuram	38863	PV