

 **Congratulations! You passed!**Grade received **28.57%** To pass 10% or higher[Go to next item](#)

Reflecting back on what you learned

Total points 14

1. Hi all!

0 / 1 point

Welcome to SIADS 521. Let's start the course by reviewing what you learned in SIADS 505. This will help you with handling data for visualizations you will create throughout this course. This will also help me understand your background knowledge.

This will not affect your grade!

This is just so I get a sense as to how sticky the items you learned in SIADS 505 are! Now, let's dig in! 🐞

Which regular expression will return all 'A's and 'B's from the string below?

```
grades="AAAABBBCCCCDDDD"
```

- ☐ [AB]
- ☒ [A][B]
- ☐ [A&B]
- ☐ {A}{B}
- ☐ {AB}

**Incorrect**

This will capture A followed by B.

2. In a regular expression, what is the special character that matches zero or more characters?

0 / 1 point

- ☒ +
- ☐ *
- ☐ ?
- ☐ #
- ☐ ^

**Incorrect**

+ captures one or more characters.

3. Which regular expression pattern would find the word 'blue' only when it is followed by 'whale' and NOT when followed by 'bottle' in the sentence "The blue whale ate the blue bottle."?

1 / 1 point

- ☐ blue .whale
- ☐ blue (.whale)
- ☐ blue (?P<whale>)
- ☐ blue [whale]
- ☒ blue (?=whale)

**Correct**

4. What will be returned after you run the regular expression below?

0 / 1 point

```
results = re.match(r'(\w+)\@(\w+),(\w+)', 'userid@umich.edu')
print(results.groups())
```

- ☒ ['userid@umich.edu'](#)
- ☐ 'umich'
- ☐ 'userid'
- ☐ 'edu'
- ☐ ('userid', 'umich', 'edu')

☒ **Incorrect**

To have this pattern with re.match, you can use '(\w+\@(\w+),\w+)'

5. Which statement is correct about the code below?

0 / 1 point

```
import pandas as pd

# code1
df = pd.DataFrame({'date': ['31/8/2021', '1/9/2021', '2/9/2021'],
                  'items': [2, 3, 4]})

# code2
df['date'] = pd.to_datetime(df['date'], dayfirst=True)
```

- ☐ After you run code1 and sort the dataframe df based on the 'date', the sequence of the values in the items column will be still 2, 3, and 4..
- ☐ After you run code1 only, the data type of the values in the date column is the integer.
- ☐ After you run code1 and code2, the data type of the values in the date columns is the object.
- ☒ Replacing code2 with df['date'] = pd.to_datetime(df['date'], dayfirst=False) would not make a difference in output.
- ☐ The output of df.shape is (2, 3).

☒ **Incorrect**

If dayfirst=False, then the "1/9/2021" and "2/9/2021" will be recognized as January 9th of 2021, and February 9th of 2021.

6. Which of the following options will merge df1 and df2 on the left into df_merged on the right?

0 / 1 point

df1				df2				df_merged					
Key	A	B		Key	C	D		Key	A	B	C	D	
0	K0	A0	B0	0	K0	C4	D4	0	K0	A0	B0	C4	D4
1	K1	A1	B1	1	K1	C5	D5	1	K1	A1	B1	C5	D5
2	K2	A2	B2	2	K4	C6	D6	2	K2	A2	B2	NaN	NaN
3	K3	A3	B3	3	K5	C7	D7	3	K3	A3	B3	NaN	NaN

- ☐ pd.merge(df1, df2, on="Key", how="outer")
- ☒ pd.merge(df1, df2, on="Key", how="inner")
- ☐ pd.merge(df1, df2, on="Key", how="right")
- ☐ pd.merge(df1, df2, on="Key", how="left")
- ☐ pd.merge(df1, df2, on="Key", how="center")

☒ **Incorrect**

With on="Key" and how="inner", there should be only row 0 and row 1 should have left.

7.

1 / 1 point

Look at the code and the first DataFrame output about cereals below. Which code should you insert to get the second DataFrame output showing means and sums of the calories per manufacturer which is represented by the column "mfr"?

```
import pandas as pd
df = pd.read_csv('cereal.csv')
df.head()
```

	name	mfr	type	calories	protein	fat	sodium	fiber	carbo	sugars	potass	vitamins	shelf
0	100% Bran	N	C	70	4	1	130	10.0	5.0	6	280	25	3
1	100% Natural Bran	Q	C	120	3	5	15	2.0	8.0	8	135	0	3
2	All-Bran	K	C	70	4	1	260	9.0	7.0	5	320	25	3
3	All-Bran with Extra Fiber	K	C	50	4	0	140	14.0	8.0	0	330	25	3
4	Almond Delight	R	G	110	2	2	200	1.0	14.0	8	-1	25	3

```
### INSERT CODE
```

	mean	sum
mfr		
A	100.000000	100
G	111.363636	2450
K	108.695652	2500
N	86.666667	520
P	108.888889	980
Q	95.000000	760
R	115.000000	920

- ☐ df.groupby("calories")["mfr"].aggregate(np.mean, np.sum)
- ☐ df.groupby("calories")["mfr"].agg((lambda x : x.mean(), lambda x : x.sum()))
- ☐ df.groupby("mfr").aggregate(("calories":np.mean, np.sum))
- ☒ df.groupby("mfr").agg(("calories":[np.mean, np.sum]))
- ☐ df.groupby("mfr").transform(("calories":[np.mean, np.sum]))



Correct

df.groupby(the column you want to use for the groupby).agg((the column you want to apply np.mean and np.sum: [np.mean, np.sum]))

8. What is the right output of the following code?

1 / 1 point

```
: import pandas as pd

numbers = {'col1': [1,1,2,3,5],
           'col2': [8,13,21,34,55]}
df = pd.DataFrame(numbers,columns=['col1', 'col2'])

df = df.apply(lambda x: x%2 == 0, axis = 1)
df
```

☒

	col1	col2
0	False	True
1	False	False
2	True	False
3	False	True
4	False	False

☐

	col1	col2
0	False	True
1	True	True
2	True	False
3	False	True

4 True True

☐

	0	1	2	3	4
col1	False	False	True	False	False
col2	True	False	False	True	False

☐

	0	1	2	3	4
col1	True	True	False	True	True
col2	False	True	True	False	True

☐

	col1	col2
0	True	False
1	True	True
2	False	True
3	True	False
4	True	True

✓ Correct

9. Which code will give you 'History' as the output, considering the Pandas Series object defined below?

1 / 1 point

```
subject_code = {1: 'Literature', 2: 'History', 3: 'Music', 4: 'Programming'}
```

```
s = pd.Series(subject_code)
```

☒ s.loc[2]

☐ s[1]

☐ s["History"]

☐ s[3]

☐ s.iloc[2]

✓ Correct

.loc gets rows or columns with particular labels, in this case that is the item with the label (not position) of 2 which is 'History'.

10. DataFrame df contains US Census data broken down by state and county. Which of the following options will change df on the left into the DataFrame on the right?

0 / 1 point

		BIRTHS2010	BIRTHS2011
STNAME	CTYNAME		
Michigan	Shiawassee County	178	692
	Tuscola County	156	506
	Van Buren County	234	943
	Washtenaw County	977	3826
	Wayne County	5918	23819
	Wexford County	109	441
Minnesota	Alitkin County	30	116
	Anoka County	1084	4068
	Becker County	102	439

Before

→

		BIRTHS2010	BIRTHS2011
STNAME	CTYNAME		
Michigan	Washtenaw County	977	3826
	Wayne County	5918	23819

After

☐ df.iloc[[('Michigan', 'Washtenaw County'), ('Michigan', 'Wayne County')]]

☐ df.loc["Washtenaw County", 'Wayne County']

☐ df.head()

☒ df[('Michigan', 'Washtenaw County'), ('Michigan', 'Wayne County')]

☐ `df.loc[('Michigan', 'Washtenaw County'), ('Michigan', 'Wayne County')]`

☒ **Incorrect**

The code will throw an error since ('Michigan', 'Washtenaw County') and ('Michigan', 'Wayne County') will be considered as two keys to look for.

11. The following figure shows the Series s. If you run the code below, what will be its output?

0 / 1 point

```
0    0.0
1    1.0
2    2.0
3    3.0
4    NaN
5    NaN
6    6.0
7    7.0
dtype: float64
```

```
s = s.fillna(method="ffill")
```

s

☒

```
0    0.0
1    1.0
2    2.0
3    3.0
4    4.0
5    5.0
6    6.0
7    7.0
dtype: float64
```

☐

```
0    0.0
1    1.0
2    2.0
3    3.0
4    3.0
5    3.0
6    6.0
7    7.0
dtype: float64
```

☐

```
0    0.0
1    1.0
2    2.0
3    3.0
4    6.0
5    6.0
6    6.0
7    7.0
dtype: float64
```

☐

```
0    0.0
1    1.0
2    2.0
3    3.0
4    NaN
5    NaN
6    6.0
7    7.0
dtype: float64
```

☐

```
0    0.0
1    1.0
2    2.0
3    3.0
4    3.0
5    6.0
6    6.0
7    7.0
dtype: float64
```

☒ **Incorrect**

.fillna(method="ffill") will fill out NaN values following values of the previous rows.

12. Which of the following options is a correct statement about the DataFrame df?

0 / 1 point

	Name	Class	Score
school1	Heru	History	86
school3	Rene	Physics	92
school2	Alex	Biology	76
school2	Yuna	Math	72

☐ `df.loc["school2"]` will give you an error since there is more than one row meeting the condition.

☐ You can have only one column as an index.

- ☒ df.drop('Name') will drop the column 'Name'.
- ☐ df.loc[:, ["Name", "Score"]] will give you all Names and Scores for all schools.
- ☐ You cannot set mixed types of data values as index labels.

☒ **Incorrect**

The default setting of .drop method is dropping a row. If you would like to drop a column, you need to set the parameter "axis=1"

13. Which of the following options is a correct statement about the code below?

0 / 1 point

```
import re
test_string = "ABCDBCBCBBBBBDE"
pattern = r"pattern"
re.findall(pattern, test_string)

: ['B', 'B', 'B', 'B', 'B', 'B', 'B', 'B']
```

- ☐ None above is correct.
- ☐ Including a non-greedy quantifier in the pattern is one way to get the output above.
- ☒ With the pattern = r'(B)*', you can get the same output.
- ☐ If re.findall(pattern, test_string) is replaced with re.finditer(pattern, test_string), you cannot get an output in a list.
- ☐ With the pattern = r'B+', you can get the output above.

☒ **Incorrect**

With * quantifier which is for zero or more, you will get spaces too for matches.

14. The df on the left side below was generated after running code 1. Then, after running code 2, the df has changed as the df on the right side below. With that said, which of the following options is an INCORRECT statement about the code and the df below?

0 / 1 point

```
import pandas as pd

### code1
df = pd.read_csv ("daily-min-temperatures.csv", parse_dates=["Date"],
                  index_col=["Date"])

### code2
df.rolling(2).mean()
```

Temp		Temp	
Date		Date	
1981-01-01	20.7	1981-01-01	NaN
1981-01-02	17.9	1981-01-02	19.30
1981-01-03	18.8	1981-01-03	18.35
1981-01-04	14.6	1981-01-04	16.70
1981-01-05	15.8	1981-01-05	15.20
...
1990-12-27	14.0	1990-12-27	14.30
1990-12-28	13.6	1990-12-28	13.80
1990-12-29	13.5	1990-12-29	13.55
1990-12-30	15.7	1990-12-30	14.60
1990-12-31	13.0	1990-12-31	14.35

3650 rows × 1 columns

3650 rows × 1 columns

- ☐ None of the above is incorrect.
- ☐ The reason why the first value of the df on the right side is NaN is because the rolling window was larger than 1 and therefore needed more values to calculate the mean.
- ☒ To calculate means for every two days as above, we can also apply resampling() to get the similar result.
- ☐ df.rolling(2).mean() could be written as df.rolling("2D").mean() since the values in the Date column are parsed as the datetime data type.
- ☐ The value 2 in the rolling(2) defines the size of rolling windows, in this case, which is every 2 rows.

⊗ Incorrect