Final Exam - DSC 10, Fall 2025

Full Name:

PID:

Seat you are in:

Instructions:

- This exam consists of 11 questions, worth a total of 141 points.
- Write your PID in the top right corner of each page in the space provided.
- Please write **clearly** in the provided answer boxes; we will not grade work that appears elsewhere. Completely fill in bubbles and square boxes; if we cannot tell which option(s) you selected, you may lose points.

  ◯ A bubble means that you should only **select one choice**.

  ☐ A square box means you should **select all that apply**.
- You may use one page of double-sided handwritten notes. Aside from this, you may not refer to any other resources or technology during the exam. No calculators!

By signing below, you are agreeing that you will behave honestly and fairly during and after this exam.

Signature:

Version A

Please do not open your exam until instructed to do so.

Important: Before proceeding, make sure to rip off the last page of this exam packet and read the data description.

Question 1 (6 pts)

For each code snippet below, determine the value of the expression on the last line.

a) (3 pts)

```
def f1(df):
    for i in np.arange(len(df.get("high_temp"))):
        if df.get("low_temp").iloc[i] < -10:
            return i
    return -10

f1(snow)
```

b) (3 pts)

```
def f2(df):
    out = np.array([])
    for k in df.index:
        snow = df.get("snowfall").iloc[-k]
        if snow > 10:
            return "snow"
        if snow < 1:
            out = np.append(out, snow)
    return out

f2(snow.sample(snow.shape[0], replace=False))
```

PID: _____

Question 2 (7 pts)

Consider the code snippet below, then answer the questions that follow.

```
def f(a, b):
    if a > 20 and b > 10:
        return "blizzard buildup"
    elif a == 0 and b > 0:
        return "fresh snowfall"
    if a > 0 and b > 0 and b <= 5:
        return "light accumulation"
    elif a > 0 and b == 0:
        return "snowy ground"
    return "no snow"

results = np.array([])
for i in np.arange(snow.shape[0]):
    output = f(snow.get("prev_snow").loc[i], snow.get("snowfall").loc[i])
    results = np.append(results, output)
```

a) (1 pt) **True or False:** The `for` loop uses the accumulator pattern.

◯ True
◯ False

b) (1 pt) **True or False:** The `for` loop could be rewritten using the Series method `.apply` with the provided function `f`.

◯ True
◯ False

c) (2 pts) Which type of plot would be most appropriate to visualize the data in `results`?

◯ Scatter plot
◯ Line plot
◯ Bar chart
◯ Histogram

d) (3 pts) Determine the value of `results[2]`.

Question 3 (11 pts)

Bianca and Ella are considering where to travel over their winter break. While they love the snow, they want to be near the coast and they get altitude sickness easily.

a) (7 pts) Fill in the blanks to make a new DataFrame, `trip`, that has a row for each date and coastal city pair, such that the `"snowfall"` column contains the snowfall for that date in that coastal city.

```
trip = snow[__(a)__].groupby(__(b)__).(__(c)__)
```

(a):

(b):

(c): **Select all** aggregation methods that can go in blank (c).

☐ `median()` ☐ `min()` ☐ `count()` ☐ `max()` ☐ `sum()`

b) (4 pts) Bianca and Ella decide to travel to the coastal city with the lowest elevation. Fill in the blanks below so that the expression correctly evaluates to the name of that city.

```
trip.__(d)___.reset_index().__(e)___.iloc[__(f)__]
```

(d):

(e):

(f):

PID: _____

Question 4 (15 pts)

Define the DataFrames `snow1` and `snow2` as follows.

```
snow1 = snow.take(np.arange(5))
snow2 = snow.take(np.arange(5, 10))
```

a) (3 pts) Calculate the TVD between the distribution of `"continent"` in `snow1` and the distribution of `"continent"` in `snow2`. Give your answer as a **simplified fraction**.

b) (3 pts) Suppose we add a new row to `snow1` and we also add this same row to `snow2`. What will the TVD between the `"continent"` distributions be after this change? Give your answer as a **simplified fraction**.

c) (3 pts) Determine the number of rows in the merged DataFrame below. Give your answer as an **integer**.

```
snow1.merge(snow2, on="continent")
```

d) (3 pts) Determine the number of rows in the merged DataFrame below. Give your answer as an **integer**.

```
snow1.merge(snow2, on="coastal")
```

e) (3 pts) Suppose that the full `snow` DataFrame has n rows, and t of these rows have a value of `True` in the `"coastal"` column. Give a **mathematical expression**, in terms of n and t, for the number of rows in the DataFrame `snow.merge(snow, on="coastal")`

Question 5 (14 pts)

Imagine that instead of having access to all of `snow`, which is quite a large DataFrame, we have only a simple random sample of 100 rows. We'll call our sample `flurry` because it's just a little bit of `snow`.

a) (4 pts) Which of the following parameters would be appropriate to estimate using a CLT-based confidence interval? **Select all that apply.**

- ☐ The median of the `"snowfall"` column of `snow`.
- ☑ The mean of the `"snowfall"` column of `snow`.
- ☐ The standard deviation of the `"snowfall"` column of `snow`.
- ☐ The largest value in the `"snowfall"` column of `snow`.
- ☐ None of the above.

CLT works for means and sums

b) (4 pts) Which of the following parameters would be appropriate to estimate using a bootstrapped confidence interval? **Select all that apply.**

- ☑ The median of the `"snowfall"` column of `snow`.
- ☑ The mean of the `"snowfall"` column of `snow`.
- ☑ The standard deviation of the `"snowfall"` column of `snow`.
- ☐ The largest value in the `"snowfall"` column of `snow`.
- ☐ None of the above.

bootstrapping works on anything that is not an "extreme" statistic (e.g. min or max)

c) (3 pts) Now we want to estimate the proportion of rows in `snow` that correspond to coastal cities, using only the data in `flurry`. In `flurry`, 70 rows correspond to coastal cities and 30 rows correspond to non-coastal cities.

The endpoints of a 95% confidence interval for this parameter can be expressed in the form

$$\frac{A}{10} \pm B\sqrt{\frac{C}{10000}}$$

where A, B, and C are all **integers**. Give the values of these integers below.

0.7 +/- 2*sqrt(0.7*0.3/100)

0.7*0.3 = 0.21

$A =$ 7 $B =$ 2 $C =$ 21

d) (3 pts) Next we want to take a new sample from `snow` that will allow us to estimate the proportion of rows in `snow` with a recorded `"snowfall"` of 0 centimeters. We want our new sample to be just large enough to guarantee that a 95% CLT-based confidence interval for this proportion is not wider than 0.1. How large of a sample should we take? Give your answer as an **integer**.

0.1 >= 4*sqrt(0.5*0.5) / sqrt(n)
sqrt(n) >= 4*sqrt(0.5*0.5) / 0.1
sqrt(n) >= 4*0.5 / 0.1
sqrt(n) >= 4*5
n >= 20^2
n >= 400

sample size = 400

PID: _____

Question 6 (13 pts)

Recall that `flurry` is a simple random sample of 100 rows from `snow`.

a) (4 pts) Fill in the blanks in the code below so that `means` evaluates to an array of 5000 bootstrapped estimates for the mean `"snowfall"` value in `snow`.

```
means = np.array([])
for i in np.arange(5000):
    resample = flurry.sample(___(a)___)
    means = np.append(means, __(b)__)
```

(a):

(b):

b) (5 pts) Imagine we were to take 1000 new samples directly from `snow`, each of size 100, and compute the mean `"snowfall"` value for each sample. Which of the following gives a reasonable approximation for the standard deviation of this set of sample means? **Select all that apply.**

- ☐ `np.std(snow.get("snowfall"))`
- ☐ `np.std(snow.get("snowfall")) / np.sqrt(snow.shape[0])`
- ☐ `np.std(snow.get("snowfall")) / 10`
- ☐ `np.std(means)`
- ☐ `np.std(means) / np.sqrt(5000)`

c) (4 pts) The following code prints a C% bootstrapped confidence interval for the mean `"snowfall"` value in `snow`, using the 5000 bootstrapped estimates stored in the array `means` and cutting off an equal amount from each tail.

```
X = ___(a)___
ci_low = np.percentile(means, X)
ci_high = np.percentile(means, 24 * X)
print([ci_low, ci_high])
```

(i.) (2 pts) What goes in blank `(a)`? In other words, what is the value of the variable `X`? Give your answer as an **integer**.

(ii.) (2 pts) What is the value of C? In other words, what is the confidence level of the interval produced? Give your answer as an **integer**.

Question 7 (17 pts)

Chicago, nicknamed the "Windy City," reaches some pretty low temperatures. Define the variable `chi_low` as follows.

chi_low is a series containing all values of low_temp for Chicago

```
chi_low = snow[snow.get("city") == "Chicago"].get("low_temp")
```

a) (3 pts) Without making any assumptions about the data in `chi_low`, determine the minimum proportion of values that lie within 3 standard deviations of the mean. Give your answer as a **simplified fraction**.

Chebyshev's Inequality: 1 - 1/z^2 = 1 - 1/3^2 = 1 - 1/9 = 8/9

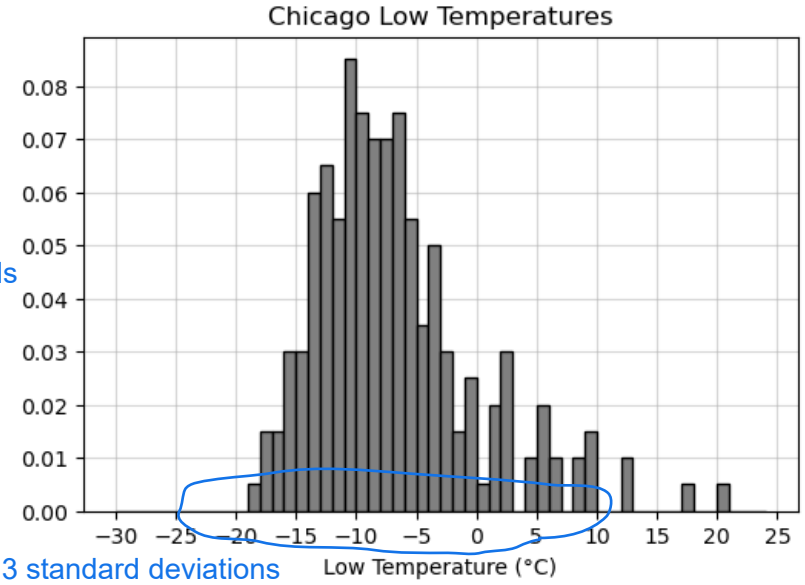
-7 - 6*3 is the lower end; -7 + 6*3 is the upper end

b) (3 pts) You are told that the mean of `chi_low` is -7°C and the standard deviation is 6°C. Define a variable `x` such that `x.mean()` calculates the actual proportion of values that lie within 3 standard deviations of the mean.

This will make x equal a series of True/False

`x =` (chi_low >= -7 - 6 * 3) & (chi_low <= -7 + 6 * 3)

c) (3 pts) You're now given a density histogram (below) with the full distribution of data in `chi_low`. Recall this data has a mean of -7°C and a standard deviation of 6°C.



The area of any bar corresponds to the proportion of values that fall into the interval captured by the bar

width of every bar appears to be 1

This is our +/- 3 standard deviations

Use the histogram to calculate the proportion of values that lie within 3 standard deviations of the mean. Your answer should be a **number between 0 and 1, to two decimal places**.

1 - (0.01*1 + 0.005*1 + 0.005*1) = 1 - (0.01 + 0.01) = 0.98

PID: _____

d) (2 pts) On the warmest Chicago day recorded in `snow`, the low temperature was 20°C. Express this temperature in standard units relative to other Chicago lows. Give your answer as an **exact decimal**.

[20 - (-7)] / 6 = 27/6 = 9/2 = 4.5

e) (3 pts) Suppose we collected four times as much data as currently depicted in the histogram above, then plotted a new histogram of the larger data set. How would we expect the standard deviation of temperatures to change?

- ◯ Decrease by a factor of 4.
- ◯ Decrease by a factor of 2.
- ◉ No substantial change.
- ◯ Increase by a factor of 2.
- ◯ Increase by a factor of 4.

NOT asking about the standard deviation of the MEAN of temperatures

f) (3 pts) What kind of distribution does the histogram depict?

- ◯ Probability distribution.
- ◉ Empirical distribution.
- ◯ Categorical distribution.
- ◯ None of the above.

Empirical distribution refers to one that is on actual data (real or simulated)

Question 8 (13 pts)

Each city in `snow` is represented many times, across multiple days. Let the *maximum snowfall* of a city be the maximum snowfall on any single day recorded for that city. If we want to compute the *average maximum snowfall* of all cities on a continent, we would first compute the maximum snowfall of every city on that continent, then take the mean of the resulting values.

Using the above definitions, you want to test the following pair of hypotheses:

- **Null:** The average maximum snowfall is the same across Asian and North American cities.
- **Alternative:** The average maximum snowfall in Asian cities is **greater than** the average maximum snowfall in North American cities.

a) (6 pts) You decide to write a permutation test for the above hypotheses. Fill in the blanks of the code below to complete the permutation test and calculate a p-value.

```
1.  cities = snow.groupby(__(a)__).__(b)__
2.  asia_na = cities[(cities.get("continent") == "Asia") |
3.                   (cities.get("continent") == "North America")]
4.
5.  def test_stat(shuffled):
6.      grouped = shuffled.groupby(__(c)__).__(d)__
7.      asia = grouped.get("snowfall").loc["Asia"]
8.      na = grouped.get("snowfall").loc["North America"]
9.      return __(e)__
10.
11. stats = np.array([])
12. for i in np.arange(10000):
13.     s = np.random.permutation(asia_na.get("snowfall"))
14.     shuffled = asia_na.assign(snowfall=s)
15.     stat = test_stat(shuffled)
16.     stats = np.append(stats, stat)
17.
18. extreme = (stats <= test_stat(cities))
19. p_value = np.count_nonzero(extreme) / 10000
```

(a): (b):

(c): (d):

(e):

b) (4 pts) You also come up with the following pair of null and alternative hypotheses:

- **Null:** There is no difference in average maximum snowfall between Asian and North American cities.
- **Alternative:** There is a difference in average maximum snowfall between Asian and North American cities.

Suppose you can change **at most one line** in the code from part (a) to test these new hypotheses. Indicate the number of the line you should change and fully rewrite that line. Or, if you believe no changes to the code in part (a) are necessary, fill in the bubble provided and leave the boxes blank.

Line number to change: [ ] *or* ◯ No lines should be changed.

The code in this line should be changed to the following:

[ ]

c) (3 pts) You successfully complete a permutation test and obtain a p-value for the hypotheses in part (b). Call this p-value p_b. Unfortunately, your computer runs out of battery before you can compute a p-value for the hypotheses in part (a). Call this p-value p_a. Can you use p_b to **approximate** p_a? If so, how?

◯ No. p_b cannot be used to approximate p_a, because the maximum snowfall for each continent cannot be assumed to have a symmetric distribution.

◯ Yes. Divide p_b in half to approximate p_a, because the difference of means is symmetric around 0 under the null hypothesis.

◯ Yes. Multiply p_b by two to approximate p_a, because the difference of means is symmetric around 0 under the null hypothesis.

◯ Yes. p_b is approximately the same as p_a, because the null hypothesis is the same in both tests.

Question 9 (12 pts)

As a resident of New York City, you suspect that your local news channel tends to overestimate the percent chance of snow in its daily weather forecast. To investigate this, you look at historical weather forecasts, and decide to focus specifically on days in which your local news channel had predicted a 40% chance of snow. With the benefit of hindsight, you can now see whether it actually snowed on 40% on those days.

The DataFrame `snow` has been queried to contain only those days that had a 40% predicted chance of snow, for New York City, stored in a new DataFrame called `snow_nyc40`. A few rows of `snow_nyc40` are shown below, but there are more rows not pictured.

	date	city	country	continent	coastal	elevation	snowfall	prev_snow	high_temp	low_temp
0	01-19-25	New York City	United States	North America	True	25	0.0	0.0	31.5	28.0
1	12-23-24	New York City	United States	North America	True	25	0.0	0.0	40.0	25.5
2	03-07-23	New York City	United States	North America	True	25	2.5	0.0	33.5	29.0

You set out to test the following hypotheses:

- **Null:** It actually snows on 40% of days for which the local news channel predicts a 40% chance of snow.
- **Alternative:** It actually snows on less than 40% of days for which the local news channel predicts a 40% chance of snow.

a) (5 pts) Fill in the blanks in the code below, which should calculate an observed test statistic, generate an empirical distribution of this test statistic under the null hypothesis, and calculate a p-value for the hypothesis test.

obs_stat will be the number of days in which it snowed at all

```
obs_stat = np.count_nonzero(snow_nyc40.get("snowfall"))
counts = np.array([])
for i in np.arange(10000):
    new_count = __(a)__
    counts = np.append(counts, new_count)
p_value = np.count_nonzero(counts __(b)__ obs_stat) / len(counts)
```

This needs to simulate days of snowfall under H_0

This is to simulate from some known categorical distribution

(a): `np.random.multinomial(snow_nyc40.shape[0], [0.4, 0.6])[0]`

(b): ◯ > ◯ >= ◯ < ◉ <= ◯ == ◯ !=

sample size of snow_nyc40

Coming from the direction of the alternative hypothesis

b) (4 pts) You want to test the same pair of hypotheses another way, using a different test statistic and rewriting the code in part (a). Which of the following could be the observed statistic for a new hypothesis test for the same pair of hypotheses?

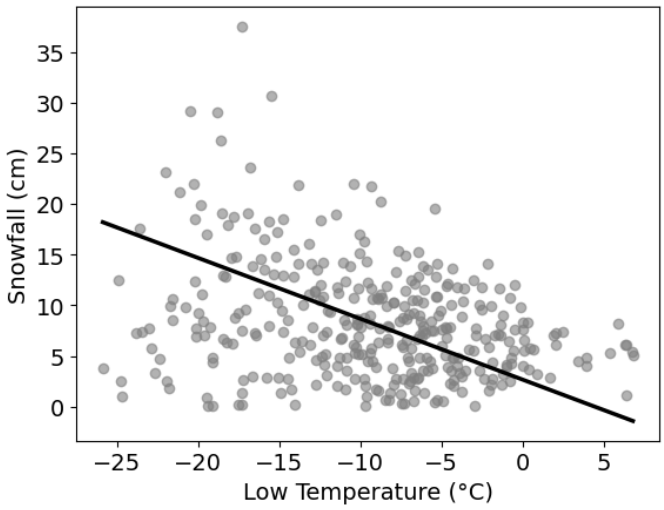
- ☐ `np.count_nonzero(snow_nyc40.get("snowfall") == 0)`
- ☐ `(snow_nyc40.get("snowfall") > 0).sum() / snow_nyc40.shape[0]`
- ☐ `(snow_nyc40.get("snowfall") > 0).mean() - 0.40`
- ☐ `(snow_nyc40.get("snowfall")).sum() / snow_nyc40.shape[0]`
- ☐ None of the above.

c) (3 pts) You also calculate a 95% CLT-based confidence interval for the proportion of days on which it actually snowed in New York City, among those for which a 40% chance of snow was predicted. Your confidence interval comes out to [0.17, 0.32]. Which of the following conclusions is correct?

- ◯ Because this confidence interval does not contain the value of 0.50, it indicates that the p-value from the hypothesis test above should be below 0.05.
- ◯ Because this confidence interval does not contain the value of 0.40, it indicates that the p-value from the hypothesis test above should be below 0.05
- ◯ Because this entire interval is above 0, it indicates that the p-value from the hypothesis test above should be above 0.05.
- ◯ There is no connection between this confidence interval and the hypothesis test because the Central Limit Theorem does not apply in this situation.
- ◯ There is no connection between this confidence interval and the hypothesis test because this confidence interval is for a proportion and the hypothesis test was performed with counts.

Question 10 (24 pts)

Pranav works for a ski resort and is curious about the relationship between the daily low temperature (°C) and snowfall (cm). He collects his own data set, recording each of these variables over a period of many days, and storing them in arrays `low_temp` and `snowfall`, respectively. He decides to use linear regression to predict `snowfall` from `low_temp`. His regression line is shown below on a scatterplot of the data.



a) (3 pts) The correlation between low temperature and snowfall at the ski resort is -0.4. The standard deviation of low temperature is 6°C and the standard deviation of snowfall is 9cm. What is the slope of the regression line predicting snowfall from temperature?

◯ -0.4
◯ -0.5
◯ -0.6
◯ -0.7

b) (3 pts) You are now told that the intercept of the regression line is 3. Which of the following is the correct interpretation of the intercept?

◯ The average snowfall in the dataset is 3cm.
◯ When the snowfall is 0cm, the predicted temperature is 3°C.
◯ When the temperature is 0°C, the predicted snowfall is 3cm.
◯ The intercept has no interpretable meaning in this context.

c) (3 pts) Estimate the value of the largest residual. Give your answer as an **integer, to the nearest multiple of 5**.

d) (3 pts) Which of the following best describes what the residual plot would look like?

◯ Residuals increase steadily as temperature increases.
◯ Residuals form a bowl shape (curved, high on both ends).
◯ Residuals have more vertical spread on the left than on the right.
◯ Residuals show no pattern and have constant vertical spread throughout.

e) (4 pts) Which of the following conclusions are valid? **Select all that apply.**

☐ The variability of the residuals changes with temperature.
☐ A curved model (such as a quadratic) would clearly fit this data better than a line.
☐ A different prediction line could eliminate the pattern seen in the errors.
☐ The linear model is appropriate, but prediction uncertainty is not constant across temperatures.
☐ None of the above.

f) (5 pts) Pranav converts `low_temp` from Celsius to Fahrenheit using the code below.

```
low_temp_f = 1.8 * low_temp + 32
```

Which of the following quantities changes as a result? **Select all that apply.**

☐ The slope of the regression line.
☐ The intercept of the regression line.
☐ The correlation coefficient.
☐ The root mean square error of the regression line.
☐ The predicted snowfall values for each day.
☐ None of the above.

g) (3 pts) Pranav performs the following bootstrapping procedure:

1. Resample the original dataset with replacement to create a bootstrap resample of the same size.
2. Fit a regression line predicting `snowfall` from `low_temp` using the bootstrap resample.
3. Calculate the sum of the slope and intercept of this regression line.
4. Repeat these steps 5,000 times, producing 5,000 values.

What does one of the 5,000 values represent?

◯ The most likely snowfall amount on a day with a low temperature of 1°C.
◯ The most likely snowfall amount on a day with a low temperature of 0°C.
◯ A predicted snowfall value at 1°C produced by one bootstrap regression line.
◯ A predicted snowfall value at 0°C produced by one bootstrap regression line.
◯ The average snowfall on all days where the low temperature was 1°C.
◯ The average snowfall on all days where the low temperature was 0°C.

Question 11 (9 pts)

The climate at UCSD in Winter 2035 looks much different than it looks today. On any given day in Winter 2035, there is a $\frac{2}{5}$ chance of precipitation. If there is precipitation, there is a $\frac{1}{2}$ chance of rain and a $\frac{1}{2}$ chance of snow (it never does both). Since rain and snow are both forms of precipitation, if there is no precipitation, it means that there is no rain or snow.

For all parts of this problem, give your answer as an unsimplified mathematical expression.

a) (3 pts) Coleman is still finishing up a few classes in Winter 2035 and is almost ready to graduate. He is planning out his week and needs to know if he should pack his mittens. In a 5-day week (Monday through Friday), what is the probability that it snows at least once?

b) (3 pts) Coleman has no class on Friday, so he doesn't have to go out on that day. What is the probability that, in a 5-day week (Monday through Friday), it snows on Friday and not on any other day?

c) (3 pts) What is the probability that, in a 5-day week (Monday through Friday), it snows on exactly one day?

PID: _____

Feel free to draw us a picture or tell us about your fondest memory from DSC 10 this quarter.

Before turning in your exam, please make sure that your PID is on every page.

Congratulations on finishing DSC 10!

❄ Snow ❄

Winter is coming, and everyone is feeling the colder weather. In the DataFrame `snow`, each row corresponds to a city-date combination representing the weather conditions in that city on a particular day. The DataFrame is sorted by date in reverse chronological order, with cities in no particular order. The columns are:

- `"date" (str)`: The date of the observation.
- `"city" (str)`: The name of the city. Each city in our data set has a different name.
- `"country" (str)`: The country in which the city is located.
- `"continent" (str)`: The continent in which the city is located.
- `"coastal" (bool)`: Whether the city is located on a coast.
- `"elevation" (int)`: The elevation of the city, measured in meters.
- `"snowfall" (float)`: The amount of snow that fell in the city on that day, measured in centimeters.
- `"prev_snow" (float)`: The amount of snow on the ground before the snow fell in the city on that day, measured in centimeters.
- `"high_temp" (float)`: The highest temperature in the city on that day, measured in degrees Celsius.
- `"low_temp" (float)`: The lowest temperature in the city on that day, measured in degrees Celsius.

The first ten rows of `snow` are shown below. The full DataFrame has many additional rows.

	date	city	country	continent	coastal	elevation	snowfall	prev_snow	high_temp	low_temp
0	12-05-2025	Chicago	United States	North America	True	179	7.5	2.0	0.0	-8.0
1	12-05-2025	Paris	France	Europe	False	48	3.0	0.0	5.0	-1.5
2	12-05-2025	Vancouver	Canada	North America	True	2	2.5	3.0	3.5	-4.0
3	12-05-2025	Beijing	China	Asia	False	43	1.0	0.0	4.5	0.0
4	12-05-2025	Cairo	Egypt	Africa	False	23	0.0	0.0	15.0	6.5
5	12-05-2025	London	United Kingdom	Europe	False	25	0.0	0.5	6.0	-2.0
6	12-05-2025	Sapporo	Japan	Asia	True	26	5.0	7.0	-4.0	-12.0
7	12-05-2025	Anchorage	United States	North America	True	31	10.5	5.0	-6.5	-10.0
8	12-05-2025	New York City	United States	North America	True	25	4.0	0.0	0.5	-1.0
9	12-05-2025	Berlin	Germany	Europe	False	34	0.0	2.0	8.0	0.0

Throughout this exam, assume that we have already run `import babypandas as bpd`, `import numpy as np`, and `import scipy`.