Question 1: foo

Arithmetic and Nested Expressions

Suppose we have the following function definition:

```python
def foo(x):
        return x + 3
```

What do the following expressions evaluate to? Assume that each expression is run independently from one another, not sequentially.

Question 1.1

```python
foo(4 * 3)
```

Answer: 15

Question 1.2

```python
foo(4) * 3
```

Answer: 21

Question 1.3

```python
foo(foo(10) // foo(-1)) % 3
```

Answer: 0

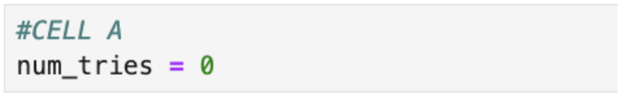
Question 2: Let’s draw triangles!

Scope

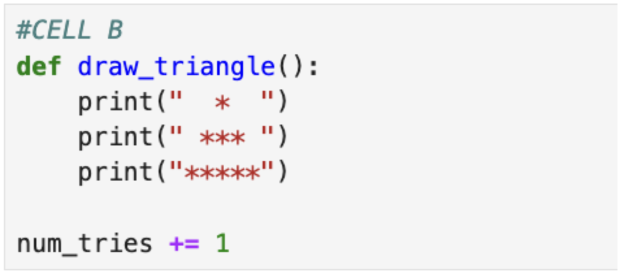
Kenneth loves triangles (they’re his favorite shape). He writes some code to draw triangles and wants to track how many times he ends up drawing one. To do this, he writes a function that draws a triangle and uses a counter to record how many times the function is called.

Given the following three workflows, which of them will correctly update the variable num_tries to reflect the number of times the function is called? Select all that apply.

Workflow 1:
Run CELL A once, then run CELL B every time draw_triangle is called.

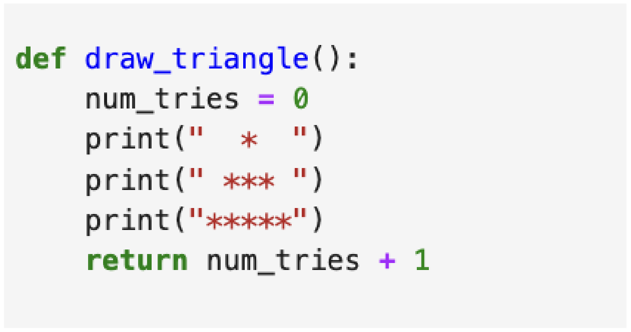


```python
#CELL A
num_tries = 0
```



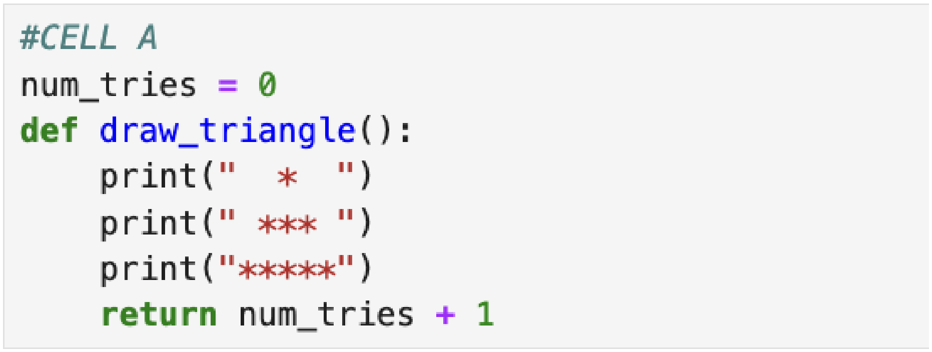
```python
#CELL B
def draw_triangle():
    print("  *  ")
    print(" *** ")
    print("*****")

num_tries += 1
```

Workflow 2:

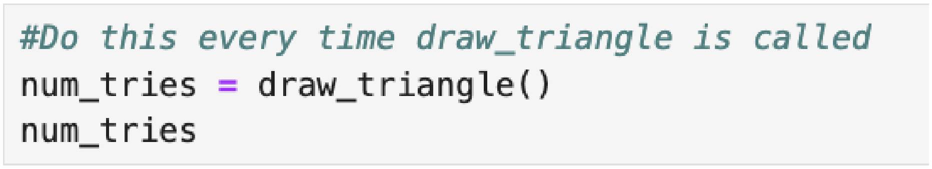


```python
def draw_triangle():
    num_tries = 0
    print("  *  ")
    print(" *** ")
    print("*****")
    return num_tries + 1
```

Workflow 3:
Run CELL A once, then every time draw_triangle is called, re-assign the return value to num_tries



```python
#CELL A
num_tries = 0
def draw_triangle():
    print("  *  ")
    print(" *** ")
    print("*****")
    return num_tries + 1
```



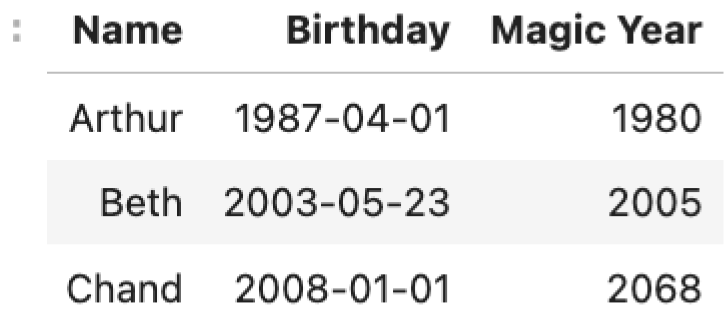
```python
#Do this every time draw_triangle is called
num_tries = draw_triangle()
num_tries
```

Answer: Implementation 1, 3

Question 3: Birthdays ! ! !

Part A: String Methods

Suppose we have a Table `birthdays`, which contains a person's name and birthday (the birthday is stored in YYYY-MM-DD format). All values are stored as Strings. You can disregard the Magic Year for now.



Name	Birthday	Magic Year
Arthur	1987-04-01	1980
Beth	2003-05-23	2005
Chand	2008-01-01	2068

In the example table above, Arthur's birthday is on April 1st, 1987.

Implement a function `age_in_given_year`, which takes in a year and a birthday, and returns the age that person will turn on that year.

For example,

```python
age_in_given_year("2010", "2000-01-01")
```

should return 10, and

```python
age_in_given_year("2010", "2010-01-01")
```

should return 0.

You may assume that the birthday is well-formatted, and that the year is always greater than or equal to the birthday year.

```python
def age_in_given_year(year, birthday):
        """Returns the age someone with a `birthday` in `year`
        Inputs:
        year: a String representing the year we care about
        birthday: a String in YYYY-MM-DD format

        Returns:
        an integer representing the age a person with the given birthday will be
        in the given year
        """
        str_year, str_month, str_day = ________________

        return ____________________________
```

SOLUTION:

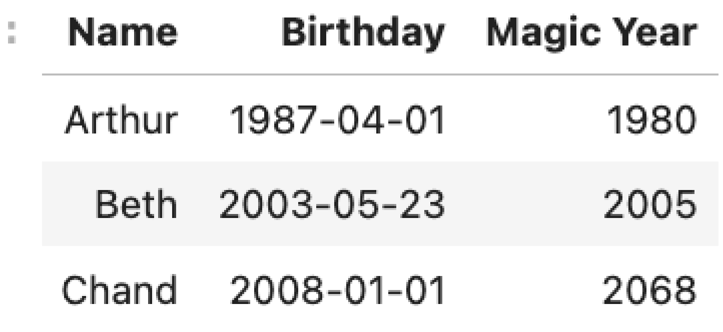
```python
def age_in_given_year(year, birthday):
        """Returns the age someone with a `birthday` in `year`
        Inputs:
        year: a String representing the year we care about
        birthday: a String in YYYY-MM-DD format

        Returns:
        an integer representing the age a person with the given birthday will be
        in the given year
        """
        str_year, str_month, str_day = birthday.split("-")

        return int(year) - int(str_year)
```

Part B: Apply and Filtering

Let’s create a new Table called “data_magic” which contains the same information as `birthdays`, but contains an extra column called “Age in Magic Year” signifying what age everyone in birthdays was in their magic year. For example, Chand’s magic year is 2068, and since their birthday is in 2008, their age in the magic year should be 60.

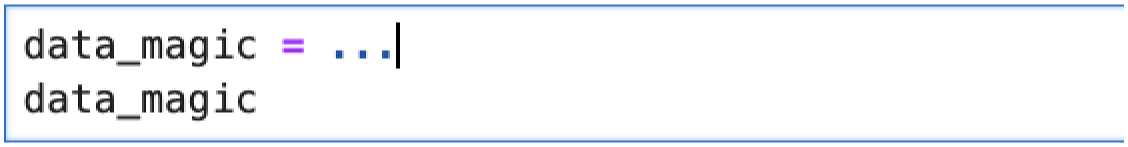
However, not everyone in our birthday table was born before their magic year! If anyone in our table was born after their magic year, they should not be a part of our resulting table.

For example, the table below

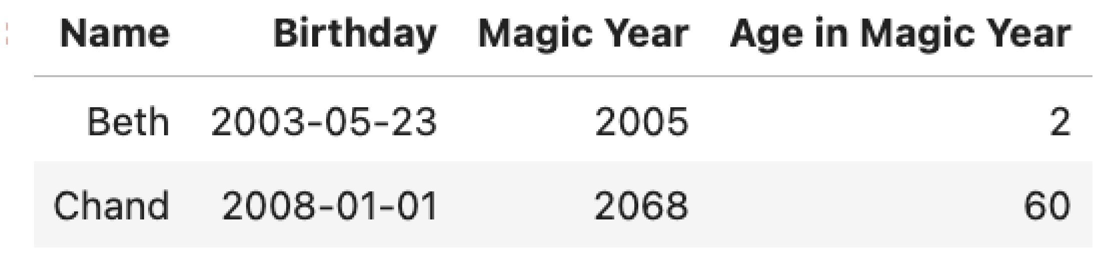


Name	Birthday	Magic Year
Arthur	1987-04-01	1980
Beth	2003-05-23	2005
Chand	2008-01-01	2068

turns into this table:



```
data_magic = ...
data_magic
```



Name	Birthday	Magic Year	Age in Magic Year
Beth	2003-05-23	2005	2
Chand	2008-01-01	2068	60

```python
data_magic = birthdays._____(blank a)_____("Age in Magic Year",
birthdays._____(blank b)_____(_____(blank c)_____,_____(blank d)_____,
_____(blank e)_____)).where("Age in Magic Year", _____(blank f)_____)
```

What goes in `_____(blank a)_____`?

1. column
2. relabeled
3. sort
4. apply
5. join
6. with_column
7. with_row

What goes in `_____(blank b)_____`?

8. column
9. relabeled
10. sort
11. apply
12. join
13. with_column
14. with_row

What goes in `_____(blank c)_____`?

15. birthdays
16. data_magic
17. age_in_given_year
18. “Name”
19. “Birthday”
20. “Magic Year”

What goes in `_____(blank d)_____`?

21. birthdays
22. data_magic
23. age_in_given_year
24. “Name”
25. “Birthday”
26. “Magic Year”

What goes in `_____(blank e)_____`?

27. birthdays
28. data_magic
29. age_in_given_year
30. “Name”
31. “Birthday”
32. “Magic Year”

What goes in blank f?

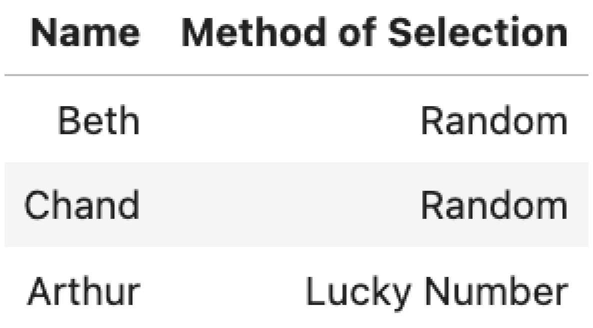
SOLUTION:

```Python
data_magic = birthdays.with_column("Age in Magic Year",
birthdays.apply(age_in_given_year, "Magic Year", "Birthday") ).where("Age in
Magic Year", are.above_or_equal_to(0))

data_magic
```

Part C: Sorting

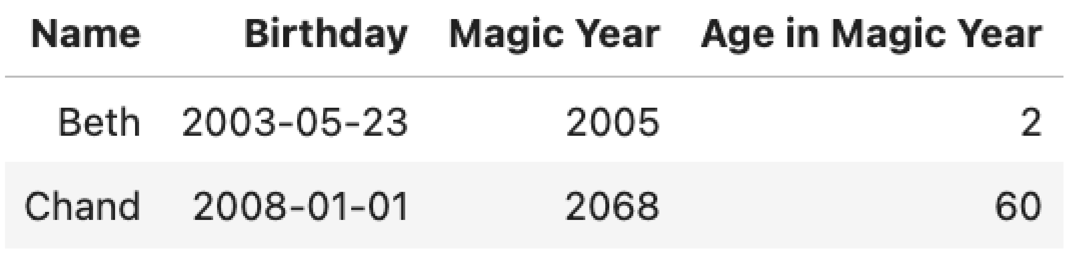
Now, assume that `data_magic` has been implemented correctly. Su Min gives us another table, `data_methods`, which notes how each person picked their Magic Year. The options are “Random”, “Lucky Number”, or “Other”.

Below is what `data_methods` looks like:



Name	Method of Selection
Beth	Random
Chand	Random
Arthur	Lucky Number

As a reminder, this is what `data_magic` looks like.

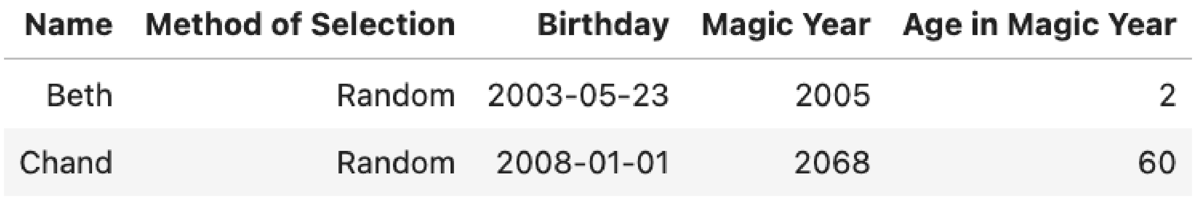


Name	Birthday	Magic Year	Age in Magic Year
Beth	2003-05-23	2005	2
Chand	2008-01-01	2068	60

1. What type of variable is the `Method of Selection`?

    a. Categorical
    b. Numerical
        i. Solution: Categorical

2. Select all following call expressions that would result in the following table:

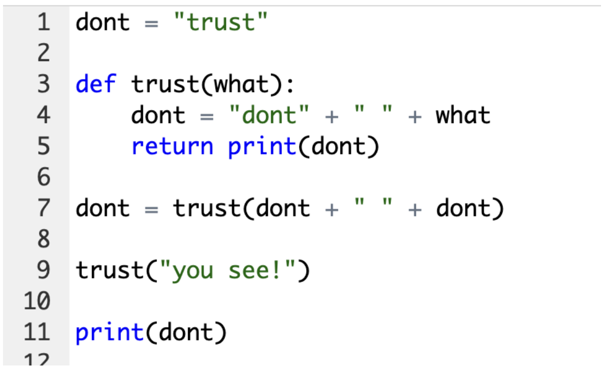


Name	Method of Selection	Birthday	Magic Year	Age in Magic Year
Beth	Random	2003-05-23	2005	2
Chand	Random	2008-01-01	2068	60

1. `data_methods.join(data_magic)`
2. `data_magic.join(data_methods)`
3. `data_methods.join(“Name”, data_magic)`
4. `data_magic.join(“Name”, data_methods)`
5. `data_methods.join(“Name”, data_magic, “Name”)`
6. `data_magic.join(“Name”, data_methods, “Name”)`
7. `data_methods.join(data_magic).sort(“Name”)`
8. `data_magic.join(data_methods).sort(“Name”)`
9. `data_methods.join(“Name”, data_magic).sort(“Name”)`
10. `data_magic.join(“Name”, data_methods).sort(“Name”)`
11. `data_methods.join(“Name”, data_magic, “Name”).sort(“Name”)`
12. `data_magic.join(“Name”, data_methods, “Name”).sort(“Name”)`
    a. Solution: 3, 5, 9, 11

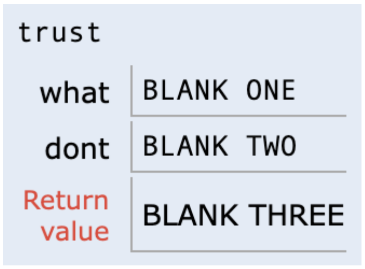
Question 4: You knew this was coming…

Given the following Python code, answer the following questions:



```
 1  dont = "trust"
 2
 3  def trust(what):
 4      dont = "dont" + " " + what
 5      return print(dont)
 6
 7  dont = trust(dont + " " + dont)
 8
 9  trust("you see!")
10
11  print(dont)
12
```

1. Besides the global frame, how many new local frames are opened?
    a. 0
    b. 1
    c. 2
    d. 3
    e. 4
        i. Solution: 2
2. Fill in the local frame that's created after line 7.



trust	
what	BLANK ONE
dont	BLANK TWO
Return value	BLANK THREE

    1. BLANK ONE should be…
        a. "dont dont"
        b. "trust trust"
        c. None
            i. Solution: b
    2. BLANK THREE should be…
        a. "dont dont"
        b. "dont trust"
        c. "dont what"
        d. None
            i. Solution: d

3. What is printed out?

4. Solution:
    a. dont trust trust
    b. dont you see!
    c. None

Question 5: Why is it SO HOT

Jedi observes that less people were out and about on October 2nd, 2024. He theorizes that it must be the heat that led people to stay indoors. It was also quite humid on that day…

Options: Confounding variable, theory, concept, causal hypothesis, associative hypothesis, exploratory research question, unit of analysis, scientific method, aggregation, disaggregation, internal validity, external validity/generalizability, categorical variable, numerical variable

Separating out the data we collect into different age groups to see if different groups reacted differently to the heat. Disaggregation

The number of people staying indoors. Numerical variable

How does temperature affect the behavior of people in urban settings? Exploratory Research Question

Jedi asks his friends what temperature it feels like in order to measure the temperature. This is a threat to… Internal Validity

Social behavior and social dynamics Concept

Jedi worries that this study only applies to students at UC Berkeley, but not students at Stanford. This is a threat to… external validity/generalizability

Humidity - Confounding variable

"Heat will impact human social behavior and social dynamics." Theory

"In order to avoid discomfort, more people stay indoors on hotter days." Causal Hypothesis

"More people stay indoors on hotter days." Associative Hypothesis