Reminders
○○

Strings Slicing
○○○○○

Split
○○○

Join
○○○○○

Adv. String Formatting
○○○○○○

Patterns (Part 1)
○○○○

Adv. Strings

David H Smith IV

University of Illinois Urbana-Champaign

Tues, Oct 4 2021

Reminders

Reminders

- Post-reading 9p1 is due Tommorow (will be posted after class).
- Homework 8 is due tommorow.
- Participation 9p1 is due Tommorow.
- Lab 4 is due Sunday after next.

Reminders
○○

Strings Slicing
●○○○○

Split
○○○

Join
○○○○○

Adv. String Formatting
○○○○○○

Patterns (Part 1)
○○○○

Strings Slicing

Reminders
○○

Strings Slicing
○●○○○

Split
○○○

Join
○○○○○

Adv. String Formatting
○○○○○○

Patterns (Part 1)
○○○○

Poll Question: Slicing

What is the result of running this code?

```
1  my_str = "CS 105"
2  print(my_str[1:2])
```

Ⓐ 'C'

Ⓑ 'CS'

Ⓒ 'CS '

Ⓓ 'S'

Ⓔ 'S '

Reminders
○○

Strings Slicing
○○●○○

Split
○○○

Join
○○○○○

Adv. String Formatting
○○○○○○

Patterns (Part 1)
○○○○

Poll Question: Slicing

What is the result of running this code?

```
1  my_str = "CS 105"
2  print(my_str[-4:-2])
```

Ⓐ 'S 1'

Ⓑ 'S 10'

Ⓒ ' 1'

Ⓓ ' 10'

Reminders
○○

Strings Slicing
○○○●○

Split
○○○

Join
○○○○○

Adv. String Formatting
○○○○○○

Patterns (Part 1)
○○○○

Poll Question: Slicing

What is the result of running this code?

```
1  my_str = "CS 105"
2  print(my_str([::2]))
```

A   'C'

B   'CS'

C   'S'

D   'C 0'

Reminders
○○

Strings Slicing
○○○○●

Split
○○○

Join
○○○○○

Adv. String Formatting
○○○○○○

Patterns (Part 1)
○○○○

Slicing

Ⓐ `string[start:stop:interval]`

Ⓑ Like range, start is inclusive stop is exclusive.

Ⓒ Interval default is 1

Ⓓ Interval is optional

Split

Poll Question: Splitting

What is the result of running this code?

```
1  my_str = "CS 105 rox"
2  result = my_str.split()
```

Ⓐ ("CS", "105", "rox")

Ⓑ ["CS 105 rox"]

Ⓒ ["CS", "105 rox"]

Ⓓ ["CS", "105", "rox"]

Poll Question: Splitting

What is the result of running this code?

```
1  csv = "1, 2, 3, 4"
2  result = csv.split(",")
```

A ['1']

B ['1, 2, 3, 4']

C ['1', ' 2', ' 3', ' 4']

D ['1,', '2,', '3,', '4']

Reminders
○○

Strings Slicing
○○○○○

Split
○○○

Join
●○○○○

Adv. String Formatting
○○○○○○

Patterns (Part 1)
○○○○

Join

Reminders
○○

Strings Slicing
○○○○○

Split
○○○

Join
○●○○○

Adv. String Formatting
○○○○○○

Patterns (Part 1)
○○○○

Poll Question: Joining

What is the result of running this code?

```
1  numlist = [1, 2, 3, 4]
2  result = ",".join(numlist)
```

A '1234'

B '1,2,3,4'

C '1, 2, 3, 4'

D TypeError

Poll Question: Joining

What is the result of running this code?

```
1 numlist = [1, 2, 3, 4]
2 result = ",".join(numlist)
```

  Ⓐ   '1234'

  Ⓑ   '1,2,3,4'

  Ⓒ   '1, 2, 3, 4'

  Ⓓ   TypeError

How do we fix this?

A Common Pattern

The generic pattern:

```
1 mylist = input_data.split(<separator>)
2 ... data processing ...
3 outputstring "<separator>".join(my_list)
```

An example of this being done on one line:

```
1 output = ",".join(string.split(",")[::2])
```

Pattern Practice

Write some code that takes a string with comma separated integers that converts the string into the square of each original value.

Go to PrairieLearn to do this problem

Pattern Practice

```python
1  def foo(numlist):
2      squaredlist = []
3      for num in numlist.split(","):
4          squaredlist.append(str(int(num) ** 2))
5      return ",".join(squaredlist)
```

Adv. String Formatting

Reminders
oo

Strings Slicing
ooooo

Split
ooo

Join
ooooo

Adv. String Formatting
o●ooooo

Patterns (Part 1)
oooo

Adv. String Formatting

```python
format_string = '{name:16}{goals:8}'

print(format_string.format(name='Player Name', goals='Goals'))
print('-' * 24)

print(format_string.format(name='Sadio Mane', goals=22))
print(format_string.format(name='Gabriel Jesus', goals=7))
```

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23

P	l	a	y	e	r		N	a	m	e							G	o	a	l	s		
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
S	a	d	i	o		M	a	n	e														
G	a	b	r	i	e	l		J	e	s	u	s											

2 2

7

```
Player Name     Goals
------------------------
Sadio Mane         22
Gabriel Jesus       7
```

Reminders
○○

Strings Slicing
○○○○○

Split
○○○

Join
○○○○○

Adv. String Formatting
○○●○○○

Patterns (Part 1)
○○○○

Alignment

For a field width of 10:

1. **Left-aligned:** `"{:<10}".format(x)`

2. **Right-aligned:** `"{:>10}".format(x)`

3. **Centered:** `"{:^10}".format(x)`

Notice the similarity between field width and how we set the number of decimals after a floating point: `"{:.2f}".format(math.pi)` $\rightarrow$ 3.14.

Reminders
○○

Strings Slicing
○○○○○

Split
○○○

Join
○○○○○

Adv. String Formatting
○○●○○○

Patterns (Part 1)
○○○○

Alignment

For a field width of 10:

1. **Left-aligned:** `"{:<10}".format(x)`
2. **Right-aligned:** `"{:>10}".format(x)`
3. **Centered:** `"{:^10}".format(x)`

Notice the similarity between field width and how we set the number of decimals after a floating point: `"{:.2f}".format(math.pi)` →3.14.

We can use a fill character to consume any unused spaces in the field width:

1. **Left-aligned:** `"{:-<10}".format(x)`
2. **Right-aligned:** `"{:->10}".format(x)`
3. **Centered:** `"{:-^10}".format(x)`

Formatting Practice

Create a function that takes a list of lists where each sub list contains 4 elements. Create and return a new list of strings where each string is composed of the four elements in each sublist and:

1. the first element is center aligned with a field with of 10
2. the second element is right aligned with a field width of 8
3. the third element is left aligned with a field width of 9
4. the fourth element is center aligned with a field width of 10 and the filler character "-".

Problem is on PrairieLearn

Example Function Call

```
1  x = [
2    ["This", "is", "a", "list"],
3    ["This", "is", "a", "list"],
4    ["This", "is", "a", "list"],
5    ["This", "is", "a", "list"],
6    ["This", "is", "a", "list"],
7    ["This", "is", "a", "list"],
8    ["This", "is", "a", "list"],
9    ["This", "is", "a", "list"],
10   ...
11 ]
12 formatted_x = formatted_str_list(x)
```

Reminders
○○

Strings Slicing
○○○○○

Split
○○○

Join
○○○○○

Adv. String Formatting
○○○○○○●

Patterns (Part 1)
○○○○

Pattern Practice

```python
1 def formatted_str_list(x):
2   formatted_strs = []
3   for a, b, c, d in x:
4     y = "{:^10}{:>8}{:<9}{:-^10}".format(a, b, c, d)
5     formatted_strs.append(y)
6   return formatted_strs
7
8 x = [
9   ["This", "is", "a", "list"],
10  ["This", "is", "a", "list"],
11  ["This", "is", "a", "list"],
12  ["This", "is", "a", "list"]
13 ]
14 formatted_x = formatted_str_list(x)
```

Patterns (Part 1)

Counting Pattern

```python
1  def count(collection):
2    counter = 0
3    for item in collection:
4      if <item meets condition>:
5        counter += 1
6    return counter
```

Computing a Sum/Total

```python
def sum(collection):
  total = 0
  for item in collection:
    total += item
  return total
```

Pattern Practice

Spend remaining class time working on last three problems in PrairieLearn.