CS 152: String Slices

CS 152: Python for STEM

Colorado State University
Computer Science Department
Slides Originally Created by Albert Lionelle and Updated by Marcia Moraes

Colorado State University

Weekly Announcements!

TODO Reminders:

- Reading 9 (zybooks) – you should have already done that ☺
- Lab 06
- Reading 10 (zybooks) – you should have already done that ☺
- Lab 07
- Reading 11 (zybooks) – you should have already done that ☺

Recall Activity

- What is string slicing? Explain with your own words.

- Write your answer in our today's attendance assignment.

String Slicing

- Slice notation has the form:
  - my_str[start:end]
- creates a new string whose value contains the characters of my_str from indices start to end - 1

```
str = "CS152 - Python for STEM"
str_Cs152 = str[0:5]
print(str)
print(str_Cs152)
```

What is printed?

What instructions should we use to parse “Python” from str?

Slicing operations

A list of common slicing operations a programmer might use.
Assume the value of my_str is 'http://en.wikipedia.org/wiki/Nasa/'

Syntax	Result	Description
`my_str[10:19]`	wikipedia	Gets the characters in indices 10-18.
`my_str[10:-5]`	wikipedia.org/wiki/	Gets the characters in indices 10-28.
`my_str[8:]`	n.wikipedia.org/wiki/Nasa/	All characters from index 8 until the end of the string.
`my_str[:23]`	http://en.wikipedia.org	Every character up to index 23, but not including my_str[23].
`my_str[:-1]`	http://en.wikipedia.org/wiki/Nasa	All but the last character.

Slice stride

- Slice notation also provides for a third argument, known as the stride. The stride determines how much to increment the index after reading each element.

```
numbers = '0123456789'

print(f'All numbers: {numbers[::]}')
print(f'Every even number: {numbers[::2]}')
print(f'Every third number between 1 and 8: {numbers[1:9:3]}')
```

```
All numbers: 0123456789
Every even number: 02468
Every third number between 1 and 8: 147
```

Field with

- Defines the minimum number of characters that must be inserted into the string.
- If the replacement value is smaller in size than the given field width, then the string's left side is padded with space characters.
- Specify by including an integer after the colon
  - {name:16}
  - specify a width of 16 characters

Align Character

Alignment type	<f-string 1> <f-string 2>	Output

Left-aligned	`f'{"Player Name":<16}{"Goals":<8}'`
	`f'{names[i]:<16}{goals[i]:<8}'`
		`Player Name Goals`
		`----------`
		`Sadio Mane 22`
		`Gabriel Jesus 7`

Right-aligned	`f'{"Player Name":>16}{"Goals":>8}'`
	`f'{names[i]:>16}{goals[i]:>8}'`
		`Player Name Goals`
		`----------`
		`Sadio Mane 22`
		`Gabriel Jesus 7`

Centered	`f'{"Player Name":^16}{"Goals":^8}'`
	`f'{names[i]:^16}{goals[i]:^8}'`
		`Player Name Goals`
		`----------`
		`Sadio Mane 22`
		`Gabriel Jesus 7`

Fill Character

- is used to pad a replacement field when the string being inserted is smaller than the field width
- default fill character is whitespace

Format specification	Value of score	Output
`{score:}`	9	`9`
`{score:4}`	9	`   9`
`{score:0>4}`	9	`0009`
`{score:0>4}`	18	`0018`
`{score:0^4}`	18	`0180`

Floating Point Precision

- indicates how many digits to the right of the decimal should be included in the output of floating types.

```
f'{1.725:.1f}'
```

```
1.7
```

```
f'{1.5:.3f}'
```

```
1.500
```

```
f'{1.666:.2f}'
```

```
1.67
```

String functions

- replace(old, new)
- find(x)
- count(x)
- split(delimiter) – if no delimiter is passed, whitespace is considered the delimiter
- join()

```
web_path = [ 'www.website.com', 'profile', 'settings' ]
separator = '/'
url = separator.join(web_path)
```

```
url = 'www.website.com/profile/settings'
```

Coding Activity

- With a Peer:
  - Write a Python function that receives the name of a csv file, reads the content of the file and calculate the sum and average of the years old data. The average should be printed with 2 decimals after the point.
  - Consider that the csv file has the following format.
- Tip: rewrite the readCSVFile function provided in Canvas.

name	years_old
Anna	18
John	17
Clara	17
Oliver	18
Julia	18
Anton	19

Coding Activity

- With a Peer:
  - Write a Python function that receives the name of a csv file, reads the content of the file and create a list with the years old that are in that file. Consider that the csv file has the format shown here.

name	years_old
Anna	18
John	17
Clara	17
Oliver	18
Julia	18
Anton	19

- Tip: rewrite the readCSVFile function provided in Canvas.
- Write a new version of your function so the list generated do not have repeated numbers.