Objectives

- Wrap up indefinite loops
- Text processing, manipulation
  - String operations, processing, methods
- Broader Issue: Self-driving cars

Review

- How do write indefinite loops in Python?
  - Why are they called indefinite loops?
- What are two ways to think about `while` loops?
- Which are more powerful: `for` loops or `while` loops?

Flipping Coins

- Problem: How many flips does it take to get 3 consecutive heads?
  - How can we simulate flipping a coin?
- Recap:
  - Have the `game` module
    - `flipCoin()` and constants for `HEADS` and `TAILS`

`game.py`
`consecutiveHeads.py`

TEXT PROCESSING

Motivation: Text Processing

- Mostly focused on numbers so far
  - A little on graphics
- We can manipulate strings to do useful work
  - Web search: finding most relevant documents to a query
  - Analyzing web logs (who is looking at my web page?)
  - Many, many others
- **Today's Focus**: the `str` data type and what you can do with them

Strings: `str`

- Used for text
- Indicated by double quotes "" or single quotes ''
  - In general, I'll use double quotes
  - Empty string: "" or ''
- Use triple quotes """ for strings that go across multiple lines

```
"""This string
is long.
Like, really, really long"""
```

STRING OPERATIONS

String Operations

Operand	Syntax	Meaning
+	`str1 + str2`	Concatenate two strings into one string
*	`str * num`	Concatenate string `num` times

- Examples:
  - `"I feel " + "sleepy"`
    - Evaluates to `"I feel sleepy"`
  - `"Oops! " * 3`
    - Evaluates to `"Oops! Oops! Oops! "`

Recall lab 0

String Comparisons

- Same operations as with numbers:
  - ==, !=
  - <, <=
  - >, >=

  } Alphabetical comparison
- Use in conditions in `if` statements

```
if courseChoice == "CSCI111":
    print("Good choice!")
else:
    print("Maybe next semester")
```

string_compare.py

Strings

- A *sequence* of one-character strings
  - Example:

  `band = “The Beatles”`

End at `len(band)`-1

characters

‘T’	‘h’	‘e’	‘ ’	‘B’	‘e’	‘a’	‘t’	‘l’	‘e’	‘s’
0	1	2	3	4	5	6	7	8	9	10

Start at 0

index or position of characters

Length of the string: 11

Built-in function: `len(string)` to find length of a string

Iterating Through a String

- Use a for loop to iterate through *characters* in a string

string of length 1

```
for char in string:
    print(char)
```

  - Read as "for each character in the string"

Python shell

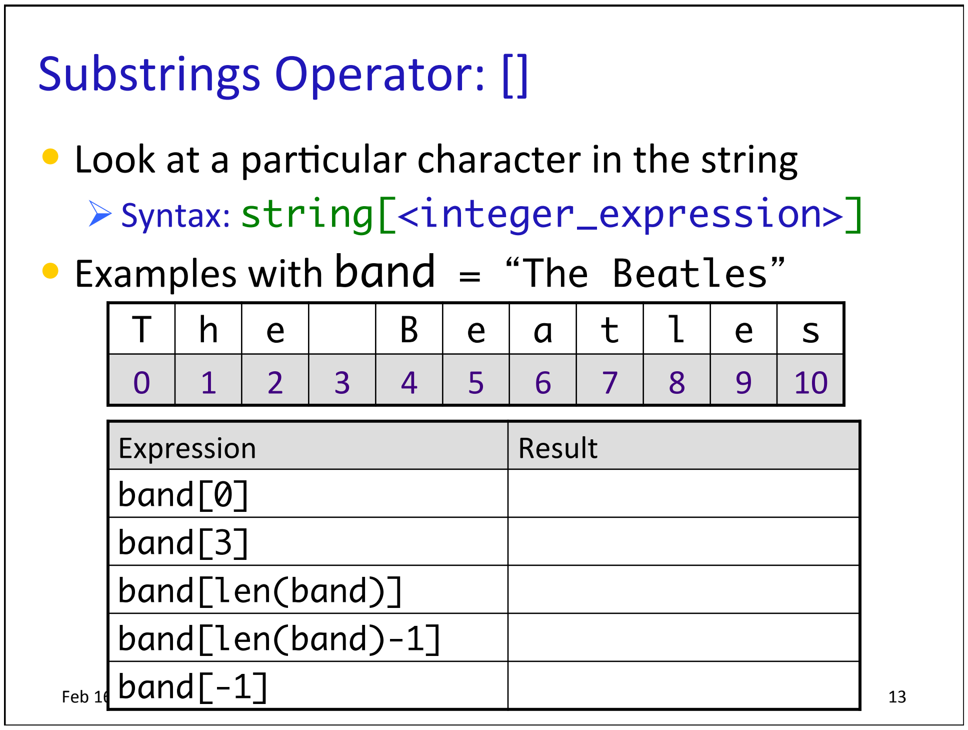
Substrings Operator: []

Literally, **not** optional

- Look at a particular character in the string
  - Syntax: `string[<integer_expression>]`
  - [Positive value]: index of character
  - [Negative value]: count backwards from end
- Examples:
  - <sequence>[0] returns the first element/char
  - <sequence>[-1] returns the last element/char

We will deal with sequences beyond strings later.

Examples in interpreter

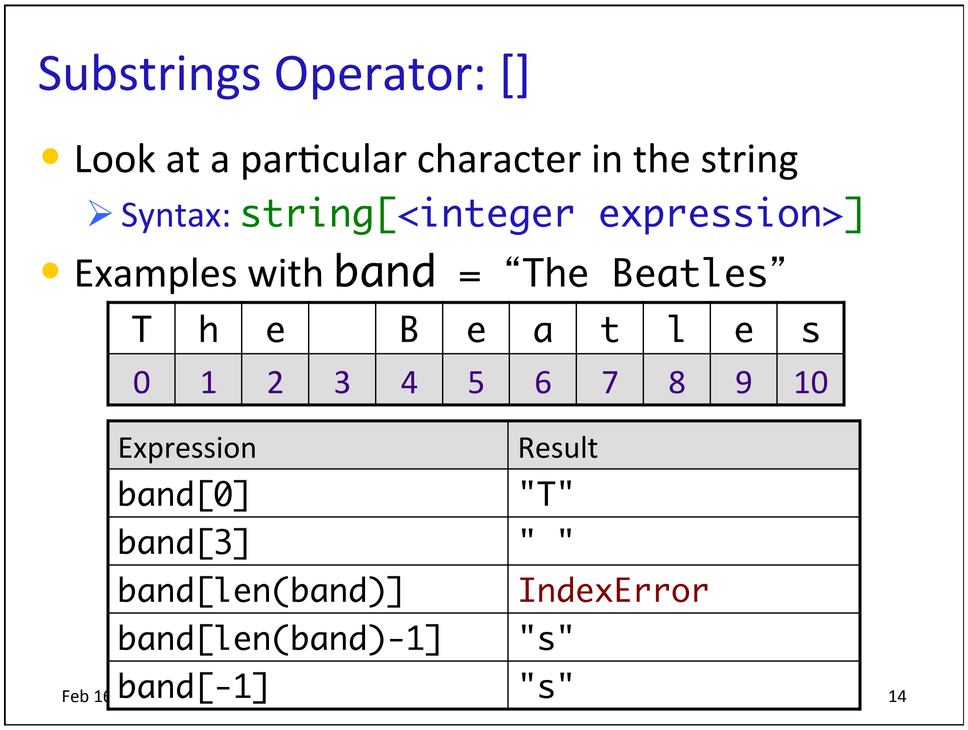


Substrings Operator: []

- Look at a particular character in the string
  - Syntax: `string[<integer_expression>]`
- Examples with band = “The Beatles”

T	h	e		B	e	a	t	l	e	s
0	1	2	3	4	5	6	7	8	9	10

Expression	Result
band[0]	
band[3]	
band[len(band)]	
band[len(band)-1]	
band[-1]	



Substrings Operator: []

- Look at a particular character in the string
  - Syntax: `string[<integer expression>]`
- Examples with band = “The Beatles”

T	h	e		B	e	a	t	l	e	s
0	1	2	3	4	5	6	7	8	9	10

Expression	Result
band[0]	"T"
band[3]	" "
band[len(band)]	IndexError
band[len(band)-1]	"s"
band[-1]	"s"

Iterating Through a String

- Alternatively, can iterate through the *positions* in a string
  - Could write as a `while` loop as well

An integer

```
for pos in range(len(string)):
    print(string[pos])
```

Index into the string

string_iteration.py

Summary: Iterating Through a String

- For each character in the string

string of length 1

```
for char in mystring:
    print(char)
```

Determines loop's behavior

- For each position in the string

An integer

```
for pos in range(len(mystring)):
    print(mystring[pos])
```

Index into the string

Substrings Operator: [:]

- Select a substring (zero or more characters) using the **[ ]** and **:**
- `<sequence>[<start>:<end>]`
  - returns the subsequence from `start` up to and **not** including **end**
- `<sequence>[<start>:]`
  - returns the subsequence from `start` to the end of the sequence
- `<sequence>[:<end>]`
  - returns the subsequence from the first element up to and **not** including **end**
- `<sequence>[:]`
  - returns a copy of the entire sequence

Substrings Operator: [:]

- Select a substring (one or more characters) using the [ ] and :
- Examples: `filename = "program.py"`

p	r	o	g	r	a	m	.	p	y
0	1	2	3	4	5	6	7	8	9

Expression	Result
`filename[0:]`	
`filename[0:2]`	
`filename[:3]`	
`filename[8:]`	
`filename[-2:]`	

Substrings Operator: [:]

- Select a substring (one or more characters) using the [ ] and :
- Examples: `filename = "program.py"`

p	r	o	g	r	a	m	.	p	y
0	1	2	3	4	5	6	7	8	9

Expression	Result
`filename[0:]`	`"program.py"`
`filename[0:2]`	`"pr"`
`filename[:3]`	`"pro"`
`filename[8:]`	`"py"`
`filename[-2:]`	`"py"`

Testing for Substrings

- Using the `in` operator
  - ➢ Used `in` before in `for` loops
- Syntax:

```
substring in string:
```

  - ➢ Evaluates to `True` or `False`
- Example:

```
if "cat" in name:
    print(name, “contains ‘cat’")
```

String Search Comparison

- What do the two `if` statements test for?

```
PYTHON_EXT = ".py"

filename = input("Enter a filename: ")

if filename[-(len(PYTHON_EXT)):] == PYTHON_EXT:
      # Appropriate output
if PYTHON_EXT in filename:
      # Appropriate output
```

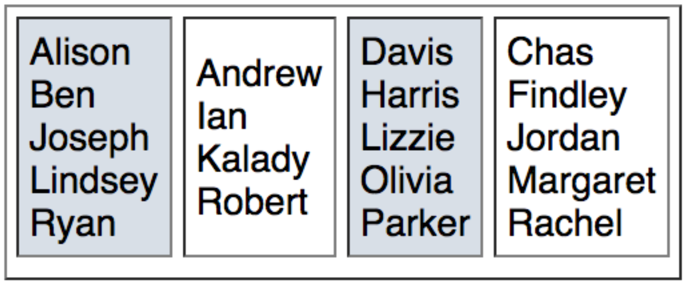
How would the program execution change if it were an **if-elif**?

search.py

Midterm Grade Calculation

- 50% - Exam 1
- 50% - Labs

Broader Issue: Self-Driving Cars



Alison	Andrew	Davis	Chas
Ben	Ian	Harris	Findley
Joseph	Kalady	Lizzie	Jordan
Lindsey	Robert	Olivia	Margaret
Ryan		Parker	Rachel

Broader Issue: Self-Driving Cars

- Self-driving cars: love ‘em or loathe ‘em
  - As a passenger?
  - As a driver (or passenger) in another car?
  - As a pedestrian?
- What are the tradeoffs of self-driving cars?
  - What guarantees about the cars would you want from the company/government?
  - Are there situations that would be particularly difficult for software to handle that a person would be better equipped to handle?
- What should the next DARPA challenge be?
- Can ethical choices be automated?

Looking Ahead

- Lab Prep Assignment: Tuesday (in progress)

Feb 16, 2018 Sprenkle - CSCI111 25