Structures: Strings

CPSC 231: Introduction to Computer Science for Computer Science Majors I
Fall 2021

Jonathan Hudson, Ph.D.
Instructor
Department of Computer Science
University of Calgary

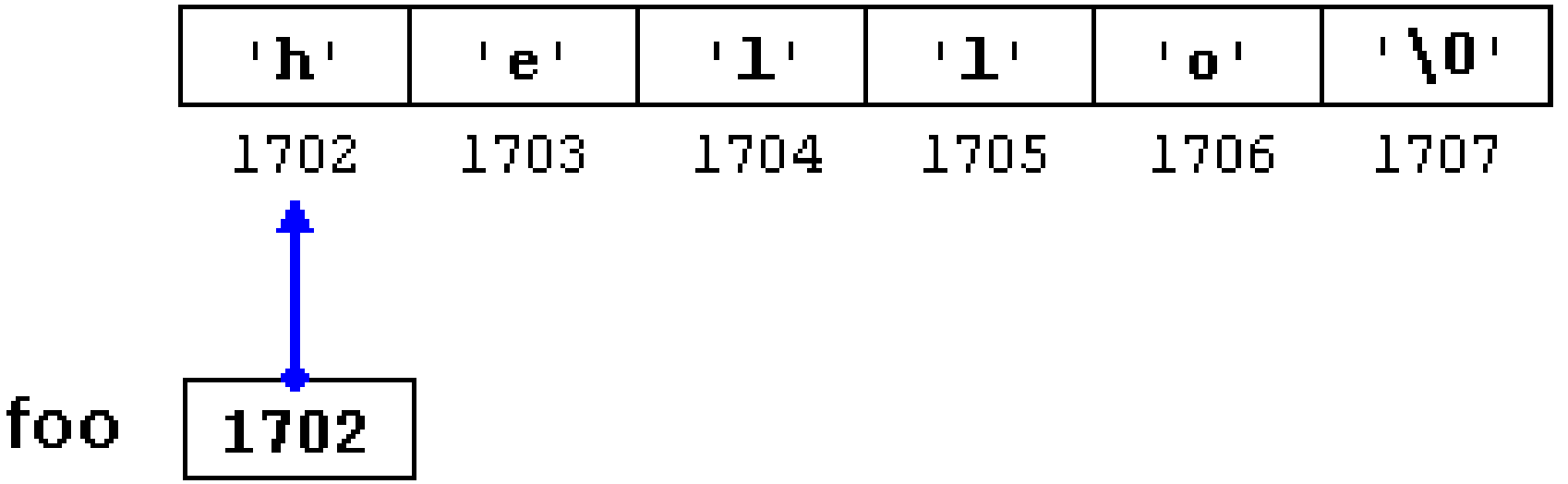
Wednesday, August 25 2021

UNIVERSITY OF CALGARY

Strings

- Strings are basically wrappers of an immutable tuple of character symbols
- In non-Python Strings are just a type of list (technically known as array C++/Java)
- Ex.

```
foo = "hello"
```



String creation

Strings

- Strings are immutable. Meaning, their values cannot change after it has been created.
- Ex.

```
greeting = "Hello, world!"
greeting[0] = 'J' # ERROR!
print(greeting)
```

Do this instead

```
greeting = "Hello, world!"
newGreeting = 'J' + greeting[1:]
print(newGreeting)
print(greeting)
```

Jello, world!
Hello, world!

Creating String Variables

- **Form:** `var = “”`
  - The string object must be enclosed with either a single quote (') or double quote (").
  - Single quote and double quote are interchangeable: “Hello World!” == ‘Hello World!’

- If your string object contains single quotes, then enclose it with double quotes, or vice versa:

```
myStr1 = “Jonathan’s lecture ROCKS!”
myStr2 = ‘The students agreed: “Yes, it does!”’
```

Creating String Variables

- Use triple single quote ('') or double quote (""") to create a string that spans multiple lines.

```
myStr3 = '''hello
        my name is Jonathan'''
```

Accessing characters

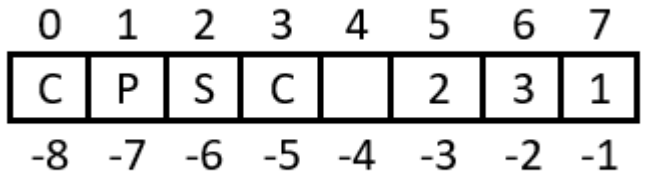
Accessing Single String Characters

- Using **indexing operator** you can access a single character from a string.
- Python uses square brackets to enclose the index

```
Course = "CPSC 231"
m = Course[2]
print(m)

lastchar = Course[-1]
print(lastchar)
```

Index



Escaping

Escaping Characters

- \\ → Backslash (keeps \)
- \’ → Single quote (keeps ’)
- \” → Double quote (keeps ”)
- \n → Newline, end of line
- \t → Horizontal tab
- …

- When Python sees the backslash (\), it treats the following character as a part of string

Escaping Characters

- Sometimes, you need to include special characters in your string, such as backslash (\), single quote, double quote, newline (\n), tab (\t), etc.

- Ex. I want to print the string *She said, "It's a beautiful morning!"* and a newline afterwards.

```
>>>print("She said, "It's a beautiful morning! "\n")
SyntaxError: invalid character in identifier
```

- backslash **** is used to indicate that the character is part of the string and to indicate the special character *newline (\n)*

```
>>>print("She said, \"It's a beautiful morning!\"\n")
She said, "It's a beautiful morning!"
```

Concatenation

String Concatenation

- Combine two or more strings together
- Example appending the last name “**Smith**” to the first name “**John**”

```
print('John', 'Smith')
print('John'+'Smith')
```

What is the difference between these calls?

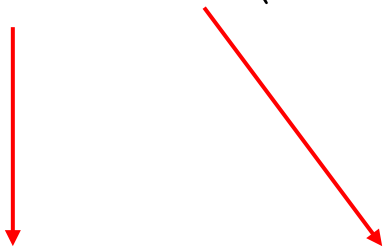
```
>>> print ('John', 'Smith')
John Smith
>>> print ('John'+'Smith')
JohnSmith
```

Formatting

String Formatting – Old-old-school way

- This original way of formatting strings:

```
>>>name = input("What is your name hooman?")
Jonathan
>>>print("Hello %s!" % (name))
```



%s a place holder of string type

% marks the start of the format specifier

```
'Hello Jonathan!'
```

String Formatting – Old-school way

- This next way of formatting strings:

```
>>>name = input("What is your name hooman?")
Jonathan
>>>print("Hello {}".format(name))
```

{} is a placeholder

Format function takes arguments that are each variable to fill into each placeholder

```
'Hello Jonathan!'
```

String Formatting – New-school way

- This new way of formatting strings:

```
>>>name = input("What is your name hooman?")
Jonathan
>>>print(f"Hello {name}"))
```

f-string

Variable name filled into {}

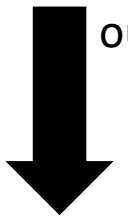
```
'Hello Jonathan!'
```

String Formatting – Float Formatting

- You can format floats to have specific width (padding) and precision.

```
width is 10
precision is 3


someValue = 12.12345
print("someValue = %10.3f" % (someValue))
```



outputs

Precision = 3

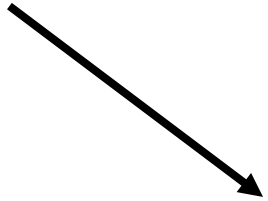
```
someValue =     12.123
```

width = 10

Operations

String Operations - Illegal

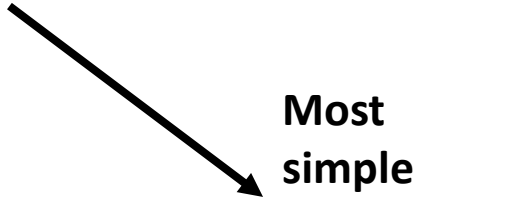
```
message - 1
"Hello" / 123
"15" + 2
```



Most simple expressions illegal

String Operations - Concatenation

```
message - 1
"Hello" / 123
"15" + 2
```



Most simple expressions illegal

```
fruit = "banana"
bakedGood = " nut bread"
print(fruit + bakedGood)
```

Legal concatentation

String Operations

```
message - 1
"Hello" / 123
"15" + 2
```

Illegal

```
fruit = "banana"
bakedGood = " nut bread"
print(fruit + bakedGood)
```

concatenation

```
print("Go" * 6)
```

GoGoGoGoGoGo

```
name = "Dinos"
print(name * 3)
```

Repetition

DinosDinosDinos

```
print(name + "Go" * 3)
```

DinosGoGoGo

```
print((name + "Go") * 3)
```

DinosGoDinosGoDinosGo

Methods

String Methods

- NOTE: A method is a function that is associated with a specific object/class.

```
ss = "Hello, World"
print(ss.upper())

print(ss.lower())
```

Outputs

```
HELLO, WORLD


hello, world
```

- Other functions include:
  - capitalize, join, strip, count, center, replace, ljust, rjust, find, rfind, …
  - For more details, refer to: https://docs.python.org/3/library/stdtypes.html

String Methods - *count()*, *strip()*, and *replace()* Examples

```
ss = "     Hello, World     "

els = ss.count("l")
print(els)

print("***" + ss.strip() + "***")
print("***" + ss.lstrip() + "***")
print("***" + ss.rstrip() + "***")

news = ss.replace("o", "***")
print(news)
```

Outputs

```
3
***Hello, World***
***Hello, World     ***
***     Hello, World***
    Hell***, W***rld
```

Slicing

String Methods - The Slice Operator

- A substring of a string is called a **slice**.

```
singers = "Peter, Paul, and Mary"
print(singers[0:5])
print(singers[7:11])
print(singers[17:21])
```

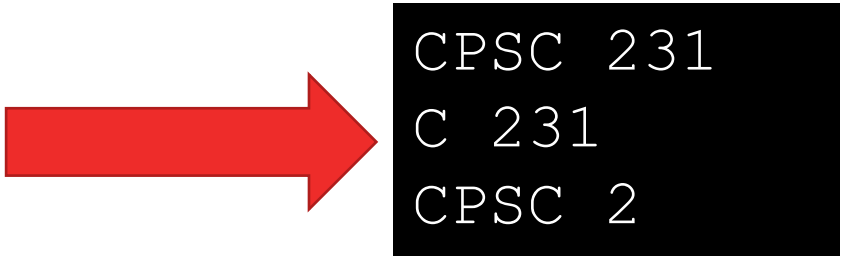
```
Peter
Paul
Mary
```

- The slice operator [n:m] returns the part of the string from the n'th character to the m'th character.
- Including the first but excluding the last.

String Methods - The Slice Operator

n and m are optional, their default values are 0 and len(str)-1, respectively

```
name = "CPSC 231"
print(name[:])
print(name[3:])
print(name[:6])
```



```
CPSC 231
C 231
CPSC 2
```

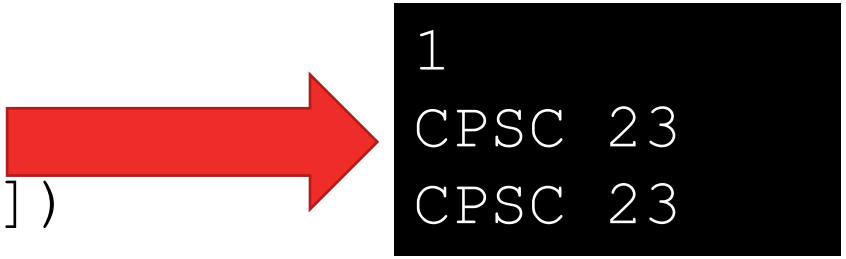
If n or m have values larger than the length of the string, no error is returned

```
name = "CPSC 231"
print(name[100:200])
print(name[:200])
print(name[100:])
```

```
''
'CPSC 231'
''
```

String Methods - The Slice Operator

```
name = "CPSC 231"
print(name[-1:])
print(name[:-1])
print(name[-8:-1])
```



```
1
CPSC 23
CPSC 23
```

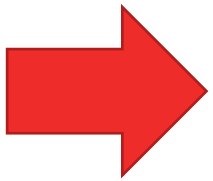
n and m can be negative values

split()

String Methods - *split()*

- The split() function takes a string argument (delimiter) returns a list of words of the original string split on this delimiter.

```
a = "this is a string"
a = a.split(" ")
print(a)
```



```
['this', 'is', 'a', 'string']
```

len()

The *len()* function

len(str) returns the number of characters in a given string.

```
fruit = "Banana"
print(len(fruit))
```

```
6
```

```
fruit = "Banana"
sz = len(fruit)
last = fruit[sz] # ERROR!
print(last)
```

```
Error! Starts from zero
```

```
fruit = "Banana"
sz = len(fruit)
last = fruit[sz-1]
print(last)
```

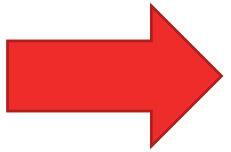
```
a
```

Examples

Example

```
def replaceSpaces(text, replacement) :
    result = ""
    for ch in text :
        if (ch == ' ') :
            result = result + replacement
        else :
            result = result + ch
    return result

print(replaceSpaces("ab ra cad abra", "*"))
```



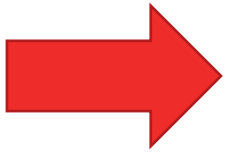
ab*ra*cad*abra

```
def countFor1(theString, theChar) :
    counter = 0
    for ch in theString:
        if (ch == theChar) :
            counter += 1
    return counter

def countFor2(theString, theChar) :
    counter = 0
    for index in range(len(theString)):
        if (theString[index] == theChar) :
            counter += 1
    return counter

def countWhile(theString, theChar) :
    counter = 0
    index = 0
    while index < len(theString) :
        if (theString[index] == theChar) :
            counter += 1
        index += 1
    return counter

print(countFor1("abracadara", "a"))
print(countFor2("abracadara", "a"))
print(countWhile("abracadara", "a"))
```



```
5
5
5
```

UNIVERSITY OF CALGARY

Onward to … file and exceptions.

Jonathan Hudson
jwhudson@ucalgary.ca
https://pages.cpsc.ucalgary.ca/~hudsonj/

UNIVERSITY OF CALGARY