

# CSc 110

## Lists of/and Strings

Adriana Picoral

# Announcements

- D2L Grades
- Final exam = 15% of final grade
- Group and Individual exams = 7.5% each

# What does `.join()` do?

- The `join()` function is a built-in function (method) for strings
- Usage:

`string.join(list)`

- Concatenates all of the elements in `list`, putting `string` in-between each element
- Can think of it as the reverse of what `split` does

# What will it print?

```
names = ["Ann", "Josephine", "Peter", "Ethan", "Alex"]  
separator = ", "  
all_names = separator.join(names)  
print("Students: " + all_names)
```

What will it print?

```
word = "coding"
```

```
print("-".join(word))
```

# What will it print?

```
chars = "a b c d a d e f a"  
char_list = chars.split(" ")  
count = char_list.count("a")  
print(count)
```

```
chars = "a b c d a d e f a"  
count = chars.count("a")  
print(count)
```

# Write the function `most_used_char`

- The function should have one parameter - a string
  - Should return the character that occurs most often in the string
  - You can use the **`count()`** function
- 
- `most_used_char('one two three')` should return `'e'`
  - `most_used_char('mississippians')` should return `'s'`
  - `most_used_char('radaration')` should return `'a'`
  - `most_used_char('zzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzz')` should return `'z'`
  - `most_used_char('at one time you are is i ')` should return `' '`

# implement swap\_words

- The **swap\_words** function should have three string parameter variables
  - **The first:** a sentence. **The second:** a word. **The third:** a word
- Should return a string that has the two words (second and third parameters) swapped
- **swap\_words('the cold night was not warm', 'warm', 'cold')**
  - Should return: **'the warm night was not cold'**
- **swap\_words('the first day after the second', 'second', 'first')**
  - Should return: **'the second day after the first'**



# What is a palindrome?

- A palindrome is a string that reads the same both forwards and backwards. Some examples of palindromes:

**'civic'**

**'radar'**

**'rotator'**

# Write the function `is_palindrome_word`

- A palindrome is a string that reads the same both forwards and backwards. Some examples of palindromes:
  - civic, radar, rotator
- `is_palindrome_word('civic')` should return `True`
- `is_palindrome_word('non')` should return `True`
- `is_palindrome_word('contemporary')` should return `False`

# Write the function `is_palindrome`

**Ignore spaces**

- `is_palindrome('otto sees otto')` should return `True`
- `is_palindrome('olson is in oslo')` should return `True`
- `is_palindrome('radar')` should return `True`
- `is_palindrome('one two three')` should return `False`
- `is_palindrome('three')` should return `False`