Chapter 10

How to work with strings

Applied objectives

1. Code, test, and debug programs that work with strings. That includes:

   slicing a string
   finding and replacing parts of a string
   splitting a string into a list of strings
   joining the items in a list into a string

Knowledge objectives

1. In general terms, describe the coding that’s used for Unicode characters.
2. Describe these built-in functions : the ord() function for working with characters and the len() function for working with strings.
3. Describe these string methods: islower(), isdigit(), startswith(), lower(), strip(), rjust(), find(), replace(), split(), and join().
4. Explain how delimiters work with the split() method and the join() method.

Two built-in functions

```
ord(char)
len(str)
```

The ordinal value of a Unicode character

```
print("5 =", ord("5"))     # 5 = 53
print("A =", ord("A"))     # A = 65
print("a =", ord("a"))     # a = 97
```

How to access a character in a string

```
message = "Hello out there!"
message[0]        # "H"
message[1]        # "e"
message[-1]       # "!"
message[16]       # IndexError: string index out of range
message[0] = "J"  # TypeError: string is immutable
```

How to slice a string

***string*[*start*:*end*:*step*]**

Examples

```
message = "Hello out there!"
message[:5]          # "Hello"
message[6:9]         # "out"
message[10:]         # "there!"
message[:-1]         # "Hello out there"
```

How to use the repetition operator (*)

```
print("=" * 20)               # ====================
print("A horse! " * 2)        # "A horse! A horse!"
```

How to use triple quotes to create a multiline string

```
query = '''SELECT categoryID, name AS categoryName
           FROM Category WHERE categoryID = ?'''
```

How to use the in keyword to search a string

Syntax

```
term in string
```

Examples

```
spam = "Congratulations. You've won a million dollars."
"million" in spam        # True
"Million" in spam        # False - search is case-sensitive
"on" in spam             # True - doesn't need to be whole word
" million " in spam      # True - uses spaces to find a whole word
" dollars " in spam      # False - ends with a period, not a space
```

Code that uses an if statement to check a search

```
search_term = input("Enter search term: ")
if search_term in spam:
    print("Term found!")
```

The console

```
Enter search term: dollar
Term found!
```

The syntax for looping through each character in a string

```
for character in string:
    statements
```

Code that prints each character in a string

```
message = "Hi!"
for char in message:
    print(char)
```

The console

```
H
i
!
```

Code that prints the ordinal value for each character

```
message = "0123 ABCD abcd"
for char in message:
    print(ord(char), end=" ")
```

The console

```
48 49 50 51 32 65 66 67 68 32 97 98 99 100
```

Basic string methods

```
isalpha()                       title()
islower()                       lstrip()
isupper()                       rstrip()
isdigit()                       strip()ljust(width)
startswith(str)                 rjust(width)
endswith(str)                   center(width)
lower()
upper()
```

How to check if a string contains all digits

```
entry = "12345"
is_integer = entry.isdigit()                    # True
```

How to check if a string starts with a substring

```
title = "The Meaning of Life"
starts_with_the = title.startswith("The")    # True
```

How to change a string to title case

```
movie = "the meaning of life"
movie = movie.title()                # "The Meaning Of Life"
```

How to strip whitespace from a string

```
ssn = "   392 55 7722  "
ssn = ssn.strip()                    # "392 55 7722"
```

How to align strings by using justification

```
print("Hammer".ljust(14), "$9.99".rjust(10))
print("Nails".ljust(14), "$14.50".rjust(10))
```

The console

```
Hammer              $9.99
Nails              $14.50
```

Four more methods of a string

```
find(str[, start][, end])
replace(old, new[, num])
removeprefix(str)
removesuffix(str)
```

Find examples

How to search for specific characters

```
email = "joel.murach@com"

at_index = email.find("@")                             # at_index = 11
dot_index = email.find(".", at_index)                  # dot_index = -1

if at_index == -1 or dot_index == -1:                  # True
    print("Invalid email address:", email)
```

How to get the first word in a string

```
title = "The Meaning of Life"
i = title.find(" ")                                    # i = 3
if i == -1:
    first_word = "This title doesn't contain a space."
else:
    first_word = title[0:i]                            # "The"
```

Replace examples

How to replace dashes with spaces in a credit card number

```
cc_number = "4012-881022-88810"
cc_number = cc_number.replace("-", " ")      # 4012 881022 88810
```

How to remove dashes from a phone number

```
phone_number = "555-555-1234"
phone_number = phone_number.replace("-", "") # 5555551234
```

Remove examples

```
email = "joel@murach.com"
```

How to remove a substring from the start of a string

```
email = email.removeprefix("joel")         # @murach.com
```

How to remove a substring from the end of a string

```
email = email.removesuffix(".com")         # @murach
```

The user interface for the Create Account program

```
Enter full name:          Eric
You must enter your full name.
Enter full name:          Eric Idle

Enter password:           sesame
Password must be 8 characters or more
with at least one digit and one uppercase letter.
Enter password:           sesaMe123

Hi Eric, thanks for creating an account.
```

The code for the Create Account program (part 1)

```
def main():
    full_name = get_full_name()
    print()

    password = get_password()
    print()

    first_name = get_first_name(full_name)
    print(f"Hi {first_name}, thanks for creating an account.")

def get_full_name():
    while True:
        name = input("Enter full name:       ").strip()
        if " " in name:
            return name
        else:
            print("You must enter your full name.")
```

The code for the Create Account program (part 2)

```
def get_password():
    while True:
        password = input("Enter password:        ").strip()
        digit = False
        cap_letter = False
        for char in password:
            if char.isdigit():
                digit = True
            elif char.isupper():
                cap_letter = True
        if digit == False or \
           cap_letter == False or \
           len(password) < 8:
            print(f"Password must be 8 characters or more \n"
                  f"with at least one digit and one uppercase "
                  f"letter.")
        else:
            return password
```

The code for the Create Account program (part 3)

```
def get_first_name(full_name):
    index1 = full_name.find(" ")
    first_name = full_name[:index1]
    return first_name

if __name__ == "__main__":
    main()
```

The split() method of a string

```
split([delimiter][, num])
```

How to split a string on whitespace

```
quotation = "These are the times that try men's souls."
words = quotation.split()
print(words[0])      # 'These'
print(words[3])      # 'times'
print(words[7])      # 'souls.'
print(words[-1])     # 'souls.'
print(words[8])      # IndexError: list index out of range
```

How to split a date on a delimiter

```
date = "11/9/1972"
date = date.split("/")
month = int(date[0])      # 11
day =   int(date[1])      # 9
year =  int(date[2])      # 1972
year =  int(date[3])      # IndexError: index out of range
```

How to split a row of data on a delimiter

```
address = "John Doe|1500 Any Street|New York|NY|10001"
address = address.split("|")
print(address[0])
print(address[1])
print(f"{address[2]}, {address[3]} {address[4]}")
```

The console

```
John Doe
1500 Any Street
New York, NY 10001
```

How to split a name into two parts

```
full_name = "Guido von Rossum"
name_parts = full_name.split(" ", 1)
print(name_parts[0])    # Guido
print(name_parts[1])    # von Rossum
```

How to join strings with the + and += operators

```
first_name = "Eric"
last_name = "Idle"
```

With the + operator

```
full_name = last_name + ", " + first_name  # Idle, Eric
```

With the += operator

```
full_name = last_name
full_name += ", "
full_name += first_name                    # Idle, Eric
```

With an f-string

```
full_name = f"{last_name}, {first_name}"   # Idle, Eric
```

The join() method of a string

```
join(sequence)
```

How to join the items of a list

```
address = ["John Doe","1500 Any Street",
           "New York","NY","10001"]
address = "|".join(address)
print(address)
```

The console

```
John Doe|1500 Any Street|New York|NY|10001
```

How to join the characters in a string

```
letters = "HORSE"
letters_spaced = " ".join(letters)
print(letters_spaced)
```

The console

```
H O R S E
```

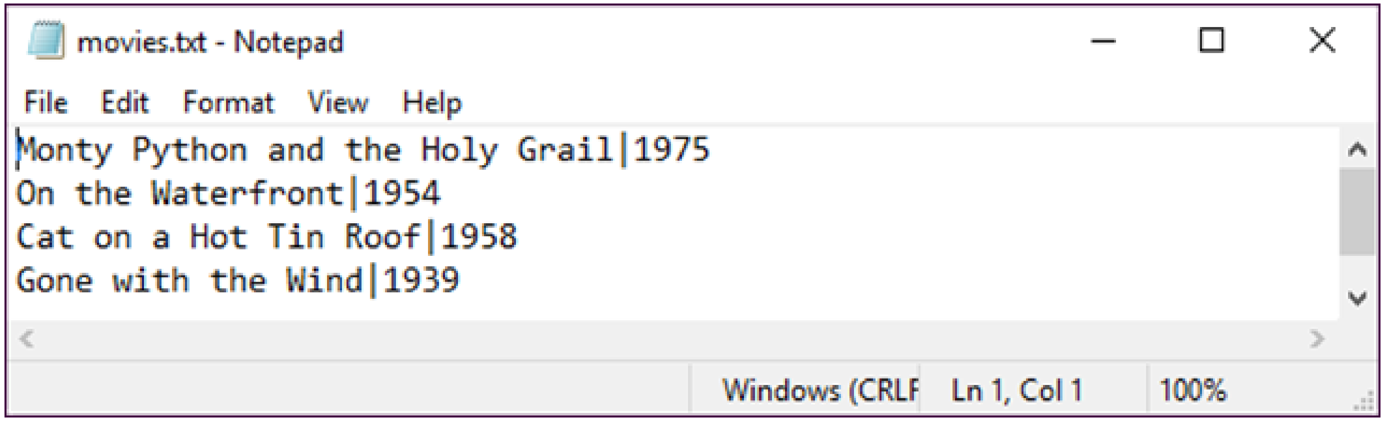
A common error when using the join() method

```
name = "John"
address = "15 E St"
full_address = name.join(address)
print(full_address)    # 1John5John JohnEJohn JohnSJohnt
```

The console

```
COMMAND MENU
list - List all movies
add -  Add a movie
del -  Delete a movie
exit - Exit program

Command: list
1. Monty Python and the Holy Grail (1975)
2. On the Waterfront (1954)
3. Cat on a Hot Tin Roof (1958)

Command: add
Name: Gone with the Wind
Year: 1939
Gone with the Wind was added.
```

The data in the text file after one record has been added to it



```
Monty Python and the Holy Grail|1975
On the Waterfront|1954
Cat on a Hot Tin Roof|1958
Gone with the Wind|1939
```

The code for the write_movies and read_movies functions

```
# a file in the current directory
FILENAME = "movies.txt"

def write_movies(movies):
    with open(FILENAME, "w") as file:
        for movie in movies:
            line = "|".join(movie)
            file.write(f"{line}\n")

def read_movies():
    movies = []
    with open(FILENAME) as file:
        for line in file:
            line = line.replace("\n", "")
            movie = line.split("|")
            movies.append(movie)
        return movies
```

The user interface for the Word Counter program

```
The Word Counter program

Number of words = 260
Number of unique words =  142
Word occurrences:
     a = 7
     above = 1
     add = 1
     ...
```

The code for Word Counter program (part 1)

```
def get_words_from_file(filename):
    with open(filename) as file:
        text = file.read()     # read str from file

    text = text.replace("\n", "")
    text = text.replace(",", "")
    text = text.replace(".", "")
    text = text.lower()

    words = text.split(" ")    # convert str to list
    words.sort()
    return words

def get_unique_words(words):
    unique_words = []
    unique_words.append(words[0])

    for i in range(1, len(words)):
        if words[i] == words[i - 1]:
            continue
        else:
            unique_words.append(words[i])
    return unique_words
```

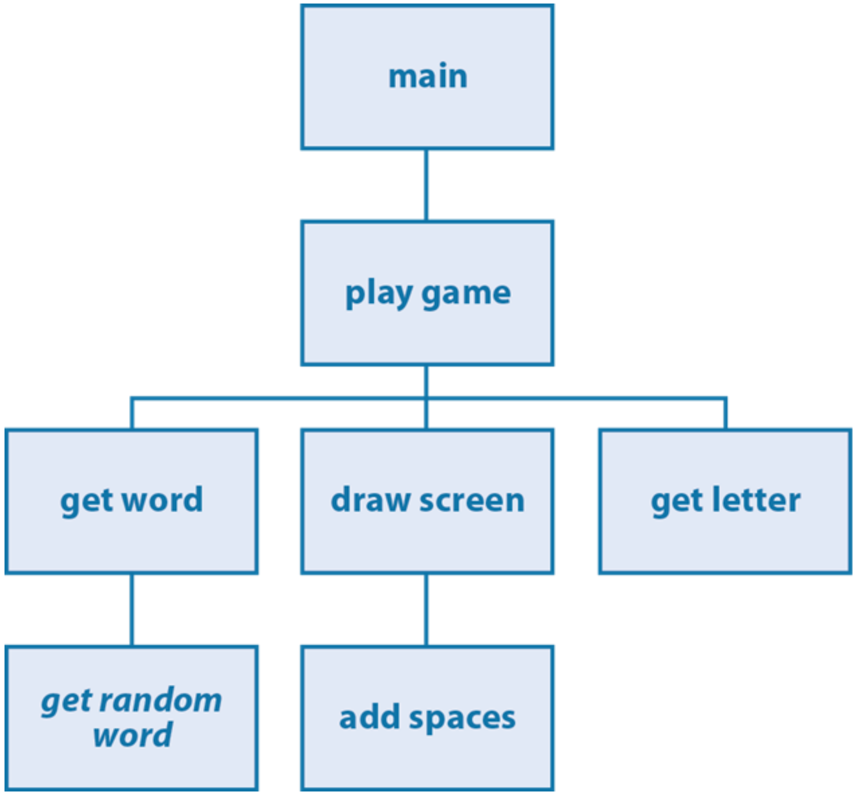
The code for Word Counter program (part 2)

```
def main():
    filename = "gettysburg_address.txt"
    print("The Word Counter program\n")

    # get words and unique words
    words = get_words_from_file(filename) # get list of words
    unique_words = get_unique_words(words)

    # display number of words and unique words
    print(f"Number of words = {len(words)}")
    print(f"Number of unique words = {len(unique_words)}")

    # display unique words and their word counts
    print("Unique word occurrences:")
    for word in unique_words:
        print(f"    {word} = {words.count(word)}")

if __name__ == "__main__":
    main()
```

The user interface for the Hangman game

```
Play the H A N G M A N game
------------------------------------------------------------------------
Word: _ _ _ _ _ _ _ _     Guesses: 0   Wrong: 0   Tried:
Enter a letter: a
------------------------------------------------------------------------
Word: _ _ _ _ _ A _ _     Guesses: 1   Wrong: 0   Tried: A
Enter a letter: e
------------------------------------------------------------------------
Word: _ _ _ _ _ A _ _     Guesses: 2   Wrong: 1   Tried: A E
Enter a letter: i
------------------------------------------------------------------------
Word: _ _ _ _ _ A _ _     Guesses: 3   Wrong: 2   Tried: A E I
Enter a letter: o
------------------------------------------------------------------------
    (The game continues)
------------------------------------------------------------------------
Word: B O U N _ A R Y     Guesses: 13   Wrong: 6   Tried: A E I O U S N C L
M R Y B
Enter a letter: d
------------------------------------------------------------------------
Word: B O U N D A R Y     Guesses: 14   Wrong: 6   Tried: A E I O U S N C L
M R Y B D
------------------------------------------------------------------------
Congratulations! You got it in  14 guesses.

Do you want to play again (y/n)?:
```

The hierarchy chart for the Hangman game



The wordlist module

```
import random

# List of words from
# http://www.free-teacher-worksheets.com/
words = [
    "aardvark",
    "air",
    ...
    ...
    "zipper",
    "zoo"
    ]

def get_random_word():
    word = random.choice(words)
    return word
```

The hangman module (part 1)

```
import wordlist

# Get a random word from the word list
def get_word():
    word = wordlist.get_random_word()
    return word.upper()

# Add spaces between letters
def add_spaces(word):
    word_with_spaces = " ".join(word)
    return word_with_spaces

# Draw the display
def draw_screen(num_wrong, num_guesses, guessed_letters,
                displayed_word):
    print("-" * 79)
    print("Word:", add_spaces(displayed_word),
          "  Guesses:", num_guesses,
          "  Wrong:", num_wrong,
          "  Tried:", add_spaces(guessed_letters))
```

The hangman module (part 2)

```
# Get next letter from user
def get_letter(guessed_letters):
    while True:
        guess = input("Enter a letter: ").strip().upper()

        # Make sure the user enters a letter and only one letter
        if guess == "" or len(guess) > 1:
            print("Invalid entry. ",
                  "Please enter one and only one letter.")
            continue
        # Don't let the user try the same letter more than once
        elif guess in guessed_letters:
            print("You already tried that letter.")
            continue
        else:
            return guess
```

The hangman module (part 3)

```
# The input/process/draw technique is common in game programming
def play_game():
    word = get_word()

    word_length = len(word)
    remaining_letters = word_length
    displayed_word = "_" * word_length

    num_wrong = 0
    num_guesses = 0
    guessed_letters = ""
    draw_screen(num_wrong, num_guesses, guessed_letters,
                displayed_word)
```

The hangman module (part 4)

```
while num_wrong < 10 and remaining_letters > 0:
    guess = get_letter(guessed_letters)
    guessed_letters += guess

    pos = word.find(guess, 0)
    if pos != -1:
        displayed_word = ""
        remaining_letters = word_length
        for char in word:
            if char in guessed_letters:
                displayed_word += char
                remaining_letters -= 1
            else:
                displayed_word += "_"
    else:
        num_wrong += 1

    num_guesses += 1
    draw_screen(num_wrong, num_guesses, guessed_letters,
                displayed_word)
```

The hangman module (part 5)

```
        print("-" * 79)
        if remaining_letters == 0:
            print(f"Congratulations! You got it in" {num_guesses} "
                  f"guesses.")
        else:
            print("Sorry, you lost.")
            print(f"The word was: {word}")

    def main():
        print("Play the H A N G M A N game")

        again = "y"
        while again.lower() == "y":
            play_game()
            print()
            again = input("Do you want to play again (y/n)?: ")

        print("Bye!")

    if __name__ == "__main__":
        main()
```