COP 3035

Intro Programming in Python

Summer 2024

Lecture 9 – part 1

Homework 3 - Due date: 06/14/2024

Lab 5 - Due Date: 06/17/2024

Exam 2 : 06/21/2024

Lecture 10 – part 2

Review

Review

For loops

Range() function

Enumerate() function

Zip() function

While loops

Break, continue, pass

Files

Integration exercise

The range function

Built-in function that generates a sequence of numbers.

Syntax:

```
range(stop)
range(start, stop)
range(start, stop, step)
```

Examples:

```
range(5) → 0, 1, 2, 3, 4
range(2, 5) → 2, 3, 4
range(2, 9, 2) → 2, 4, 6, 8
```

- **Question:** How do you “cast” a range object to a list in Python?

The enumerate function

Built-in function that returns **an iterator** yielding pairs (index, element) for each element in a sequence.

Syntax:

```
   enumerate(iterable, start=0)
```

Example:

```
for index, value in enumerate(['a', 'b', 'c']):
    print(index, value)
```

The zip function

Built-in function that **aggregates items** from two or more iterables.

Syntax:

```
zip(*iterables)
```

Example:

```
names = ["Alice", "Bob", "Charlie"]
scores = [85, 92, 88]
for name, score in zip(names, scores):
    print(name, score)
```

while loops

- While loops continue to execute a block of code **while** some condition remains **True**.

Syntax of the while loop:

```
while some_condition:
    # Do something
else:
    # Do something different
```

break, continue, pass

break – Breaks out the current closest enclosing loop.

continue – Goes to the top of the closest enclosing loop.

pass – Does nothing at all.

Files

Opening and closing files

```
myFile = open('myFilename','w+')
myFile.close()
# open file arguments: 'r', 'w', 'w+', 'a+'
```

Reading and writing

`myFile.read():` Read the entire file at `offset` position.

`myFile.seek(offset):` Change files current position

`myFile.readlines():` To read the file line by line

`myFile.write('text'):` To write a string to the file

A safer way to open the file

To automatically close a file when done:

```
With open('myFilename', mode='r') as myfile:
    # Perform file operations
```

File open modes

Mode	Description
'r'	Open for reading (default). Error if the file does not exist.
'w'	Open for writing. Truncates to zero length or creates a new file.
'x'	Open for exclusive creation. Fails if the file already exists.
'a'	Open for writing, appending at the end if the file exists.
'b'	Binary mode. Use with other modes for binary files (e.g., **'rb'**, **'wb'**).
't'	Text mode (default). Use with other modes for text files (e.g., **'rt'**, **'wt'**).
'+'	Update mode, for reading and writing (e.g., **'r+'**, **'w+'**, **'a+'**).

- **'r+’** : Open for reading and writing without truncating the file. The pointer is at the beginning.
- **‘w+’** : Truncates the file to zero length or creates a new one for reading and writing.
- **‘a+’** : Open for reading and writing. Appends at the end. Creates the file if it doesn't exist.

Lecture 10 – part 3

Integration Exercise

Exercise: Favorite songs report

```
MY FAVORITE SONGS
--------------------------------------------------------------------------------
Genre                    Artist                   Song Title               Duration
--------------------------------------------------------------------------------
Pop                      Adele                    Lady Antebellum          3:48
Electro-pop/dance        LMFAO                    *Party Rock Anthem*      4:32
Hip hop/pop              Nicki Minaj              Super Bass               3:20
Indie pop                Foster The People        Pumped Up Kicks          4:00
Country                  Lady Antebellum          Just A Kiss              3:38
--------------------------------------------------------------------------------
The total album duration is: 19 minutes and 18 seconds
--------------------------------------------------------------------------------
```

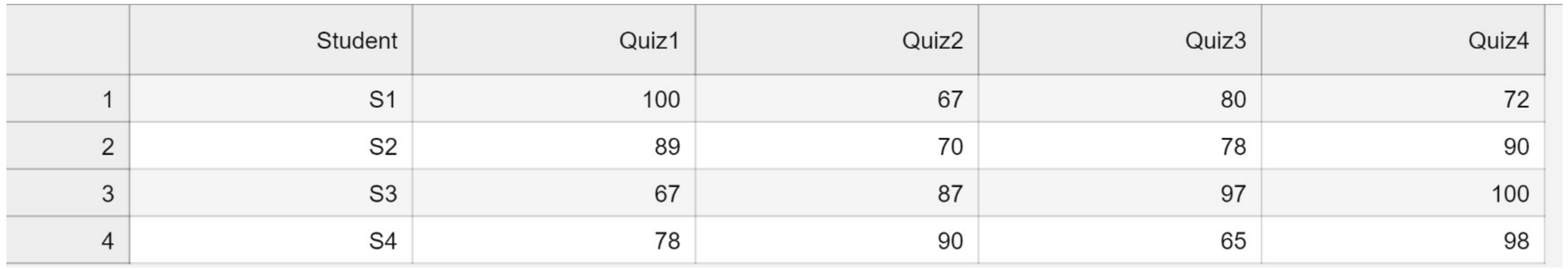
Plan:

1. Create a dictionary for every song.
2. Organize the songs in a list (playlist).
3. Set a variable for the favorite song.
4. Iterate over the playlist, print each song, flag the favorite and do computations.
5. Compute the total duration and the final report.

Lecture 10 – part 4

Integration Exercise

`class.csv`



	Student	Quiz1	Quiz2	Quiz3	Quiz4
1	S1	100	67	80	72
2	S2	89	70	78	90
3	S3	67	87	97	100
4	S4	78	90	65	98

Grade Conversion Table

Score	Letter	Score	Letter	Score	Letter	Score	Letter	Score	Letter
93-100	A	85-89	B+	75-79	B-	68-71	C	50-59	D
90-92	A-	80-84	B	72-74	C+	60-67	C-	0-49	F

Results

```
-------------------------------------------------------------
Student          Quiz1       Quiz2       Quiz3       Quiz4


-------------------------------------------------------------
S1                 A           B           C+          C+
S2                 B+          B-          A-          A-
S3                 C-          A           A           A
S4                 B-          C-          A           A
```

Plan:

1. Create a table of grade equivalences as a **dictionary**, **using tuples as keys for intervals**.

2. Open the file and store each line (representing a student) in a list of strings.

3. Iterate over the grade list; use enumerate() to obtain the index for each line.

4. For each grade, **check if it falls within a specified interval tuple**.

5. Append the corresponding grade equivalences to the result list.

6. Print the results.