

## HOMEWORK 8

### 1. LIST COMPREHENSION

Please use list comprehension to solve the following problems.

#### Problem 1.

- (1) Given a list of words, create a new list containing the first letter of each word. For example, for the list

```
words = ["apple", "banana", "cherry"]
```

the output should be

```
["a", "b", "c"]
```

- (2) Given a list of words, create a new list that contains only the words that start with a vowel (a, e, i, o, u). For example, for the list

```
words = ["apple", "banana", "orange", "grape", "umbrella"]
```

the output should be

```
["apple", "orange", "umbrella"]
```

#### Problem 2.

- (1) Given a list of tuples, create a new list containing the second element from each tuple. For example

```
tuples = [(1, "a", 1), (2, "b", 5), (3, "c", 4)]
```

the output should be

```
["a", "b", "c"]
```

- (2) Given a list of tuples where the first element is a number, create a new list that includes only those tuples where the first element is greater than a specified threshold value. For example

```
tuples = [(1, "a"), (3, "b"), (5, "c")]
```

```
threshold = 2
```

the output should be

```
[(3, "b"), (5, "c")]
```

## 2. TUPLES

### Problem 3.

- (1) Problem: Create a tuple containing the names of four of your favorite fruits. For me, it is

```
favorite_fruits = ("mango", "pear", "grape", "cherry")
```

- (2) Print out the first and last elements of your tuple.
- (3) Add a new fruit to this tuple.
- (4) With this updated tuple, create a list of fruits whose names have at least 5 characters. You can use the append method and/or list comprehension for this problem.

**Problem 4.** Write a function named `flatten_tup(nested_tuple)` that takes a nested tuple as input and returns a single flattened tuple. For example

```
nested_tuples = ((1, 2), (3, 4), (5, 6))
flatten_tup(nested_tuples)
```

should return

```
(1, 2, 3, 4, 5, 6)
```

For this problem, you need to initiate an empty tuple before the for loop and keep updating this tuple. To create an empty tuple, you can use the syntax

```
new_tup = ()
```

## 3. DICTIONARIES

**Problem 5.** Let us consider the following dictionary

```
state_capitals = {
    "Illinois": "Springfield",
    "California": "Sacramento",
    "Texas": "Austin",
    "New York": "Albany",
    "Florida": "Tallahassee"
}
```

In this dictionary, each key represents a state, and the corresponding value is its capital.

- (1) For each state in the dictionary, print out a sentence in the format: “The capital of Illinois is Springfield”.
- (2) Add the key-value pair “Wisconsin”: “Madison” to this dictionary.
- (3) Using the updated dictionary, create a list of states whose capital names start with the letter A.

**Problem 6.** Write a function named `get_average_score(student_grades)` that takes in a dictionary `student_grades(d)` as input and returns a new dictionary containing each student's name and their average score rounded to the one decimal place. Here `student_grades` is a dictionary whose each key is a student's name and the corresponding value is a list of student's scores throughout the semester.

For example

```
student_grades = {
    "Benjamin": [56, 79, 90, 22, 50],
    "David": [88, 62, 68, 75, 78],
    "Becky": [95, 88, 92, 85, 85],
    "Anne": [76, 88, 85, 82, 90],
    "Isis": [99, 92, 95, 89, 99]
}
```

```
get_average_score(student_grades)
```

should return

```
{"Benjamin": 59.4, "David": 74.2,
"Becky": 89.0, "Anne": 84.2, "Isis": 94.8}
```

For this problem, you can use the `sum/max/min/len` etc methods for a list.

**Problem 7.** Write a function named `highest_rating(books_and_ratings)` that takes a dictionary `books_and_ratings` as input and returns the list of books that have the highest rating. The dictionary `books_and_ratings` contains the titles of books and their corresponding ratings on a scale from 1-5. For example

```
books_and_ratings = {
    "Siddhartha": 4.8,
    "1984": 4.8,
    "To Kill a Mockingbird": 4.7,
    "Pride and Prejudice": 4.6,
    "The Great Gatsby": 4.8
}
```

```
print(highest_rating(books_and_ratings))
```

should return

```
[ "Siddhartha", "1984", "The Great Gatsby"]
```