Name: _____

1. Dictionary Tracing

For each part, write exactly what the code prints. If the code prints a dictionary or a list, include all brackets, commas, colons, and quotation marks exactly as Python would display them.

a. What will the following code print?

```
scores = {"Alicia": 8, "Brandon": 5, "Camila": 10}

scores["Brandon"] = scores["Brandon"] + 3
scores["Daniel"] = 7

print(scores["Brandon"])
print(scores["Daniel"])
print(scores)
```

b. What will the following code print?

```
inventory = {"pencils": 12, "notebooks": 4, "erasers": 7}

inventory["pencils"] -= 5
inventory["notebooks"] += 2

item = "erasers"
inventory[item] = inventory[item] + 1

print(inventory["pencils"])
print(inventory["notebooks"])
print(inventory[item])
```

2. Dictionary Fundamentals

Complete each line of code using a dictionary operation.

Suppose we begin with the following dictionary:

```
student = {
    "name": "Maya",
    "age": 16,
    "grade": 11
}
```

a. Write an expression that accesses `student`'s age.

b. Write one line of code that changes `student`'s grade to `12`.

c. Write one line of code that adds a new key `"favorite color"` with the value `"purple"`.

d. Write an expression that checks whether the key `"email"` appears in the dictionary.

3. Update Inventory

Write a function `addItem(inventory, item, amount)` that updates an inventory dictionary.

The dictionary maps the name of an item to the number of that item currently available.

- If `item` is already a key in `inventory`, add `amount` to its current value.
- If `item` is not already a key, add it to the dictionary with the value `amount`.
- Return the updated dictionary.

Examples:

- `addItem({"apple": 4, "banana": 2}, "apple", 3)` returns `{"apple": 7, "banana": 2}`
- `addItem({"apple": 4}, "orange", 5)` returns `{"apple": 4, "orange": 5}`

Complete the function below.

```
def addItem(inventory, item, amount):
    if ________________________________________:
        ____________________________ = ____________________________
    else:
        ____________________________ = ____________________________
    return ________________________________________
```

4. Counting Birds

Write a function `countBirds(birds)` that accepts a list of bird names and returns a dictionary containing the number of times each bird appears.

Each bird name should be a key in the dictionary, and its value should be the number of times that bird appears in the list.

Examples:

- `countBirds(["parrot", "crow", "parrot"])` returns `{"parrot": 2, "crow": 1}`
- `countBirds(["owl", "owl", "owl"])` returns `{"owl": 3}`
- `countBirds([])` returns `{}`

Complete the function below.

```
def countBirds(birds):

    counts = ______________________________

    for bird in ___________________________:

        if _________________________________:

            ___________________________ = ___________________________

        else:

            ___________________________ = ___________________________

    return ________________________________________
```

5. Average Friend Age

A dictionary called `friends` maps each friend's name to their age.

Write a function `averageFriendAge(friends)` that returns the average age of all friends in the dictionary.

You may assume that the dictionary contains at least one friend.

Examples:

- `averageFriendAge({"Ava": 15, "Noah": 17})` returns `16.0`
- `averageFriendAge({"Mia": 14, "Leo": 16, "Zoe": 18})` returns `16.0`

Complete the function below.

```
def averageFriendAge(friends):

    total = ______________________________

    for name in ___________________________:

        total = ___________________________________

    return ____________________________________
```

6. Cafe Orders

A cafe stores its menu in a dictionary where each key is the name of a drink and each value is its price.

```
menu = {"tea": 3, "coffee": 4, "smoothie": 6, "water": 2}
```

Write a function `orderTotal(menu, order)` that accepts the menu dictionary and a list of drink names. The function should return the total price of all drinks in the order.

You may assume that every drink in `order` appears as a key in `menu`.

Examples:

- `orderTotal(menu, ["tea", "coffee"])` returns `7`
- `orderTotal(menu, ["smoothie", "smoothie", "water"])` returns `14`
- `orderTotal(menu, [])` returns `0`

Complete the function below.

```
def orderTotal(menu, order):

    total = ______________________________

    for drink in __________________________:

        total = ___________________________________

    return ____________________________________
```

7. Group Students

Write a function `groupStudents(students)` that accepts a dictionary mapping student names to their assigned group number.

The function should return a new dictionary mapping each group number to a list of the students assigned to that group.

Example:

```
students = {
    "Amari": 1,
    "Bianca": 2,
    "Carlos": 1,
    "Dalia": 3,
    "Evan": 2
}
```

The call `groupStudents(students)` should return:

```
{
    1: ["Amari", "Carlos"],
    2: ["Bianca", "Evan"],
    3: ["Dalia"]
}
```

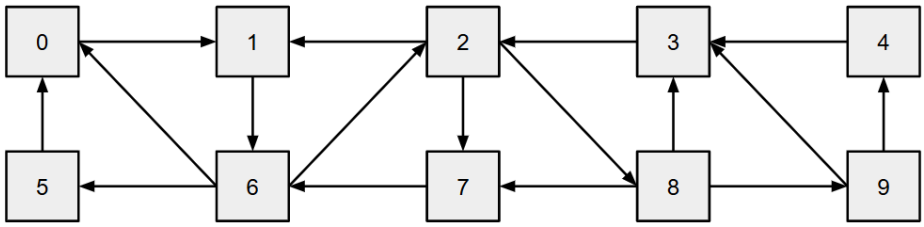
Complete the function below.

```
def groupStudents(students):

    groups = ______________________________

    for name in ___________________________:

        group = _________________________________

        if ______________________________________:

            groups[group] = ____________________________

        groups[group].append(___________________________)

    return ________________________________________
```

8. Graph Traversals

This problem was adapted from Princeton University's COS 226 course.

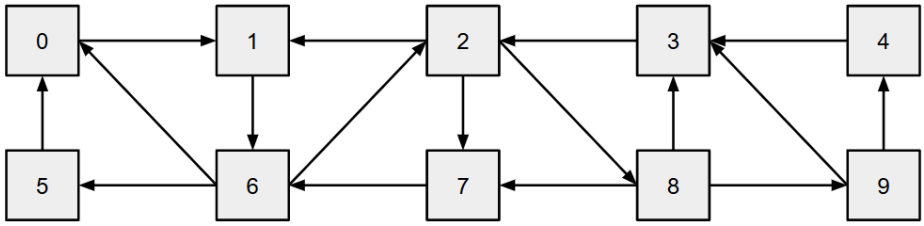
Suppose we have the following graph below.



a. Run depth-first search on the graph above, starting at vertex `0`. Break ties by selecting the vertex with the smaller integer value.

DFS Order: _____ _____ _____ _____ _____ _____ _____ _____ _____ _____

The graph is given again for ease of reference.



b. Run breadth-first search on the graph above, starting at vertex `0`. Break ties by selecting the vertex with the smaller integer value.

BFS Order: _____ _____ _____ _____ _____ _____ _____ _____ _____ _____