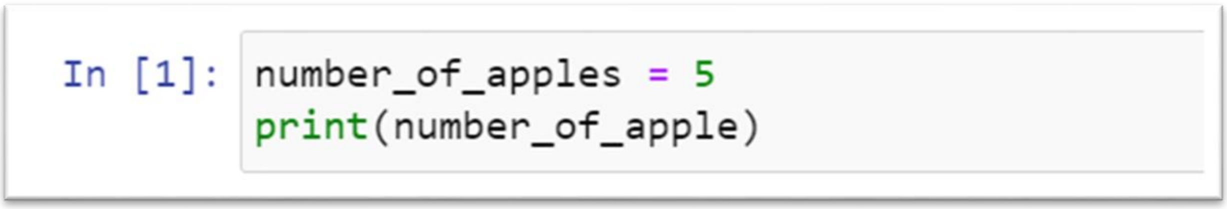
Task 1: Review and Fix Errors in Code

-submitted by Alishna Mukhya A

- **<u>Code Snippet 1:</u>** Variable Name Typo

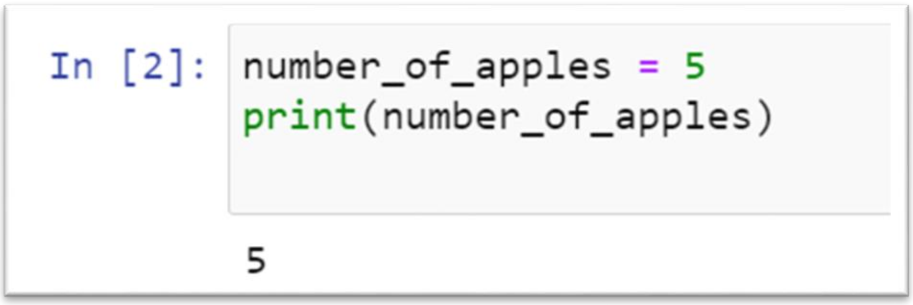
Original Code:



```
In [1]: number_of_apples = 5
        print(number_of_apple)
```

Error: The error occurred because the variable `number_of_apple` is not defined. There seems to be a typo in the variable name.

Corrected Code:

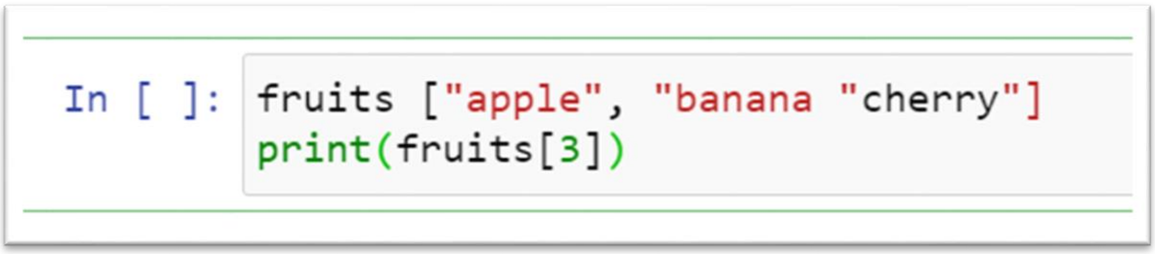


```
In [2]: number_of_apples = 5
        print(number_of_apples)

        5
```

Explanation: The code encountered an error because it tried to use a variable called number_of_apple, which doesn't exist. To resolve this, the correct variable name number_of_apples should be used consistently throughout the code. The error was fixed by ensuring that the correct variable name with the "s" at the end (number_of_apples) was used in the print statement.

- **<u>Code Snippet 2</u>:** Accessing List Elements Out of Range

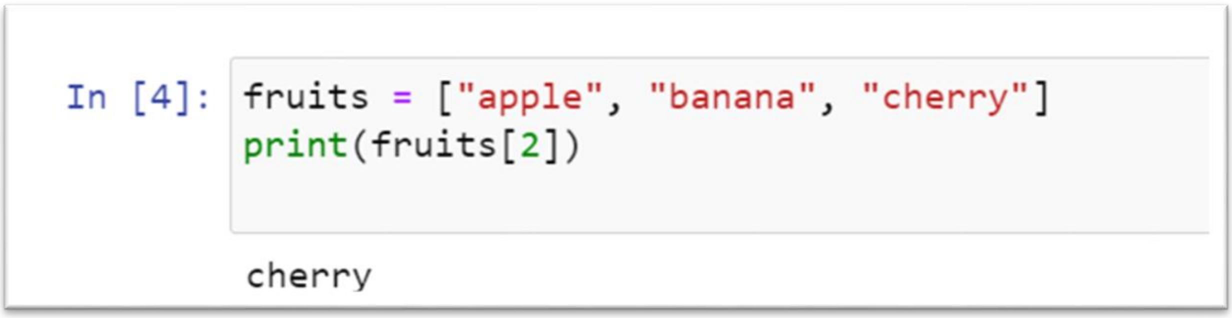
Original Code:



```
In [ ]: fruits ["apple", "banana "cherry"]
        print(fruits[3])
```

Error: The error occurred because the code attempted to access an index in a list that does not exist. This means the index specified is beyond the range of the list.

Corrected Code:



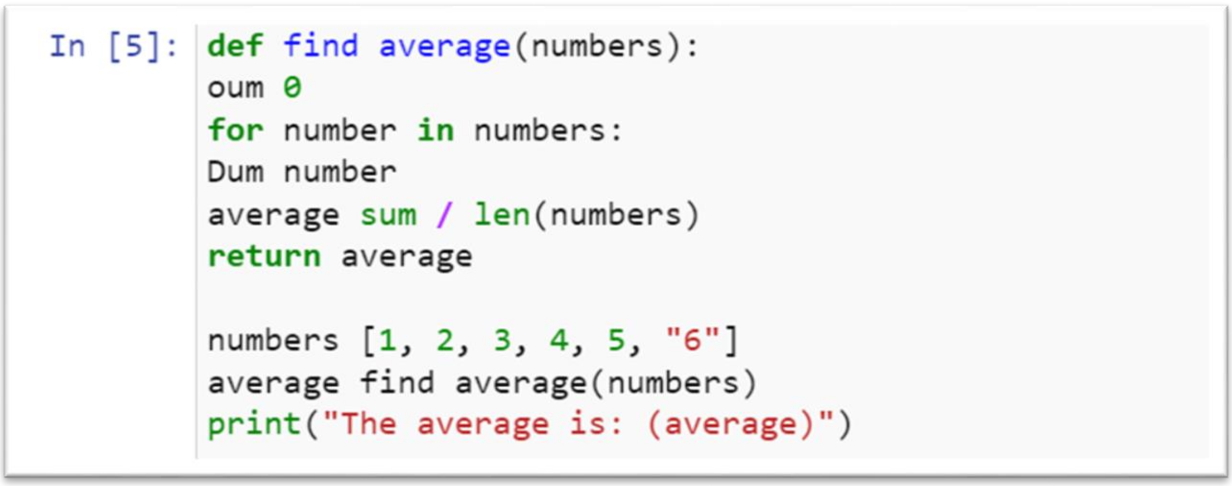
```
In [4]: fruits = ["apple", "banana", "cherry"]
        print(fruits[2])

        cherry
```

Explanation: The error occurred because the code attempted to access an index in a list that does not exist. In Python, list indices start at 0. For example, if you have a list called `fruits` with three elements ("apple", "banana", and "cherry"), the valid indices are 0, 1, and 2. Accessing the third element would be done with `fruits[2]`.

Additionally, a syntax error in the list was corrected by adding the equal sign (`=`) and a missing comma between "banana" and "cherry".

- **<u>Debugging Exercise 3:</u>** Function Not Behaving as Expected

Original Code:

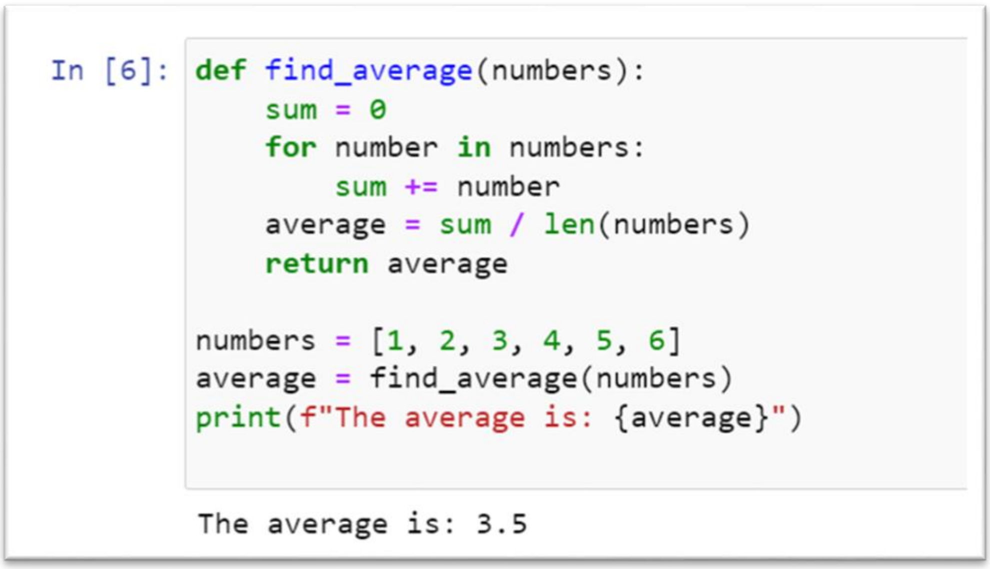


```
In [5]: def find average(numbers):
        oum 0
        for number in numbers:
        Dum number
        average sum / len(numbers)
        return average

        numbers [1, 2, 3, 4, 5, "6"]
        average find average(numbers)
        print("The average is: (average)")
```

Error: The errors included SyntaxError and TypeError due to various issues in the code:

1. **SyntaxError**: There were multiple syntax errors in the code, such as incorrect function names and indentation problems.
2. **TypeError**: This occurred because a string ("6") was included in the list of numbers, which should contain only integers or floats.

Corrected Code :



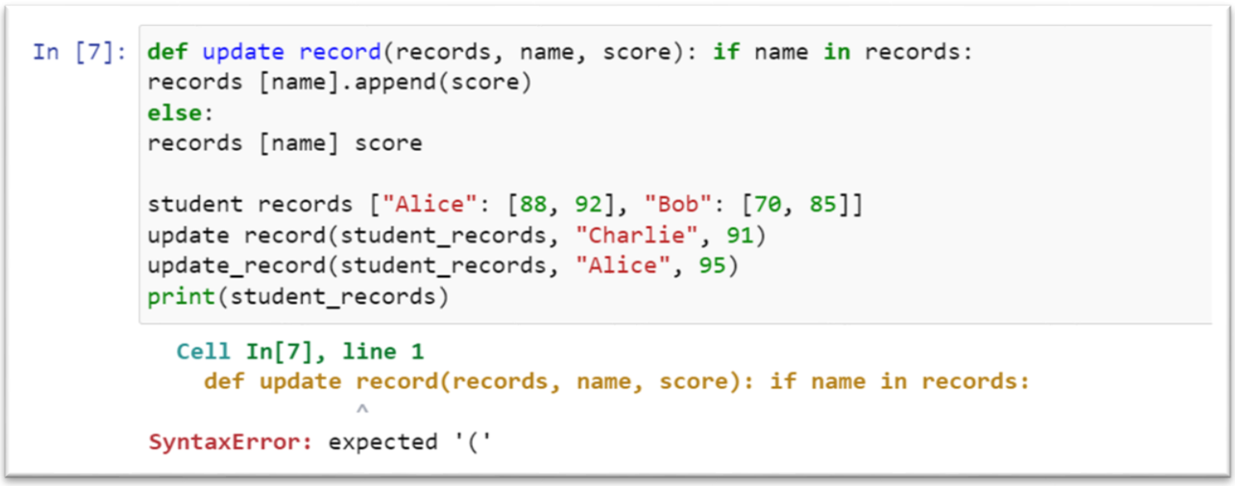
```
In [6]: def find_average(numbers):
            sum = 0
            for number in numbers:
                sum += number
            average = sum / len(numbers)
            return average

        numbers = [1, 2, 3, 4, 5, 6]
        average = find_average(numbers)
        print(f"The average is: {average}")

        The average is: 3.5
```

Explanation:

- The function name was corrected to find_average.
- Variable names like sum were fixed to avoid conflicts with built-in functions.
- Indentation and syntax errors were corrected throughout the code.
- The string "6" was removed from the list numbers.
- The += operator was used to accumulate the sum of numbers.
- An f-string was used to correctly format and print the average.

- **Exercise 4:** Incorrect Dictionary Usage

Original Code:

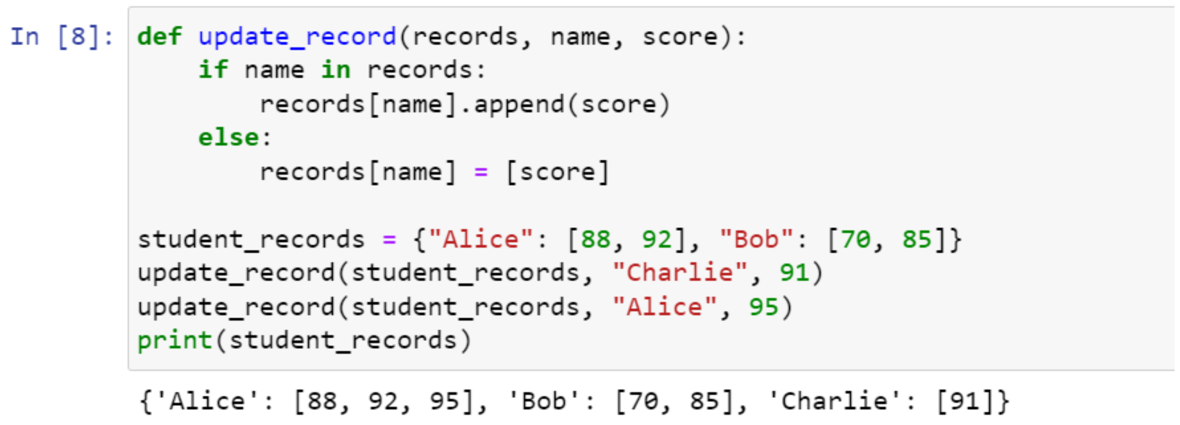


```
In [7]: def update record(records, name, score): if name in records:
        records [name].append(score)
        else:
        records [name] score

        student records ["Alice": [88, 92], "Bob": [70, 85]]
        update record(student_records, "Charlie", 91)
        update_record(student_records, "Alice", 95)
        print(student_records)

          Cell In[7], line 1
            def update record(records, name, score): if name in records:
                       ^
        SyntaxError: expected '('
```

Error:

- **Incorrect Dictionary Assignment**: This error typically arises when there's an attempt to access or modify a dictionary in a way that doesn't conform to Python's dictionary operations.
- **Syntax Errors**: These are caused by invalid Python syntax, such as missing colons in dictionary definitions or incorrect usage of dictionary methods.

Corrected Code :



```
In [8]: def update_record(records, name, score):
            if name in records:
                records[name].append(score)
            else:
                records[name] = [score]

        student_records = {"Alice": [88, 92], "Bob": [70, 85]}
        update_record(student_records, "Charlie", 91)
        update_record(student_records, "Alice", 95)
        print(student_records)

        {'Alice': [88, 92, 95], 'Bob': [70, 85], 'Charlie': [91]}
```

Explanation:

- Corrected the function name to ‘update_record’.
- Fixed the dictionary assignment by using ‘records[name] = [score]’ for new keys.
- Corrected the dictionary syntax by using ‘=’.
- Fixed the function calls to match the corrected function name.