Q1.

Code Snippet 1: Variable Name Typo
Code:

```python
number_of_apples = 5
print(number_of_apple)
```

Error:

The error in the above code is NameError : name ‘number_of_apple’ is not defined

Corrected Code:

main.py

```python
number_of_apples =  5
print(number_of_apples)
```

```
5
```

Error Explanation:

A NameError occurs in Python when a variable or name is used in code without being defined. This can happen if there's a typo in the variable name or if the variable hasn't been assigned a value before it's used.

In this code, number_of_apple is mistakenly used instead of number_of_apples, leading to a NameError because number_of_apple hasn't been defined. To fix this error, ensure that variable names are correctly spelled and defined before use.

Q2.

Code Snippet 2: Accessing List Elements Out of Range
Code:

```python
fruits = ["apple", "banana", "cherry"]
print(fruits[3])
```

Error:

The error in the above code is IndexError : list index out of range

Corrected code to print last element in the list:

```python
fruits = ["apple","banana","cherry"]
print(fruits[2])
```

```
cherry
```

Error Explanation:

In Python, list indices start from 0. This means that if we have a list with three elements, the indices for those elements would be 0, 1, and 2.

Even though the length of the list is 3, we access the last element using index 2 because indices start from 0 and go up to n-1, where n is the length of the list.

Q3.

Debugging Exercise 3: Function Not Behaving as Expected

```python
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}")
```

Error:

The error in the above code is

TypeError: unsupported operand type(s) for +=: ‘int’ and ‘str’

Correct Code:

```python
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 error is a TypeError. This error occurs because the code attempts to add a string to an integer, which is not a valid operation in Python. The list numbers contains a mix of integers and a string "6". When the find_average function iterates through this list and tries to add each element to the sum, it encounters "6" which is a string. Python does not allow adding a string to an integer using the += operator, resulting in a TypeError. To fix this error, we need to ensure that all elements in the list are of the same data type (preferably numerical) before performing arithmetic operations on them.

Q4.

Exercise 4: Incorrect Dictionary Usage
Code:

```python
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)
```

Error:

The error you can expect is an 'AttributeError'. This error occurs because the code attempts to append a score to a value that is not a list (or iterable), causing an attribute-related issue.

Correct Code:

main.py

```python
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)
```

input

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

Error Explanation:

The error raised as the Charlie is not present in the given dictionary and when we add a new value from second time to the key we can't append the value as its type was a integer not a list so to fix the error we have change the code in else block in the function i.e to records [name]=[score]