ASSIGNMENT 1

Q1. Create one variable containing following type of data:

(i) string: my_string = "Hello, world!"

(ii) list: my_list = [1, 2, 3, 4, 5

(iii) float: my_float = 3.14

(iv) tuple: my_tuple = (1, 2, 3, 4, 5)

Question2-

The data types of the given variables are as follows:

(i) var1: string

(ii) var2: string

(iii) var3: list

(iv) var4: float

Q3. Explain the use of the following operators using an example:

(i) /: It performs division between two numbers.

(ii) %: It returns the remainder of the division of the left operand by the right operand.

(iii) //: It performs division and returns the integer part of the quotient.

(iv) **: Exponentiation Operator

Question4:

```
my_list = [10, 3.14, "Hello", True, [1, 2, 3], {'a': 1, 'b': 2}, (4, 5), None, 5+3j, False]

for element in my_list:
    print(f"Element: {element}, Data Type: {type(element)}")
```

Q5. Using a while loop, verify if the number A is purely divisible by number B and if so then how many times it can be divisible.

```
A = int(input("Enter the number A: "))

B = int(input("Enter the number B: "))

divisions = 0

while A % B == 0:
    A /= B
    divisions += 1

if divisions > 0:
    print(f"{A} is purely divisible by {B}, and it can be divided {divisions} times.")
```

```
else:
    print(f"{A} is not purely divisible by {B}.")
```

Question 6: Create a list containing 25 int type data. Using for loop and if-else condition print if the element is divisible by 3 or not.

```
# Create a list containing 25 integer type data
my_list = list(range(1, 26))
# Iterate over each element in the list
for num in my_list:
    # Check if the element is divisible by 3
    if num % 3 == 0:
        print(f"{num} is divisible by 3")
    else:
        print(f"{num} is not divisible by 3")
```

Q7. What do you understand about mutable and immutable data types? Give examples for both showing this property.

In Python, data types are categorized into mutable and immutable based on whether their values can be changed after they are created.

Mutable Data Types: Mutable data types allow for in-place modifications after creation. This means that you can change the value of the object without creating a new object. Examples of mutable data types in Python include lists, dictionaries, and sets.

```
# Example of a mutable data type: list
my_list = [1, 2, 3]
print("Original list:", my_list)
# Modify the list in-place
my_list[0] = 100
print("Modified list:", my_list)
```

Immutable Data Types: Immutable data types, on the other hand, do not allow their values to be changed after creation. If you try to modify an immutable object, a new object will be created with the updated value. Examples of immutable data types in Python include integers, floats, strings, tuples, and frozensets.

```
# Example of an immutable data type: string
my_string = "hello"
print("Original string:", my_string)
```

```
# Attempt to modify the string (this will raise an error)
# my_string[0] = 'H'
# Instead, we can create a new string with the modified value
new_string = my_string.upper()
print("New string:", new_string)
```