.Ans 1)Variables containing different types of data:

```
# (i) string

string_variable = "Hello"


# (ii) list

list_variable = [1, 2, 3, 4, 5]


# (iii) float

float_variable = 3.14


# (iv) tuple

tuple_variable = (1, 2, 3, 4, 5)
```

Q2. Data types of given variables:

```
# (i) var1

# Data type: string


# (ii) var2

# Data type: string


# (iii) var3
```

Data type: list

(iv) var4

Data type: flow

Q3. Explanation of operators:

- `/`: Division operator. It divides the left operand by the right operand.
- `%`: Modulus operator. It returns the remainder of the division operation.
- `//`: Floor division operator. It performs division where the result is rounded down to the nearest integer.
- `**`: Exponentiation operator. It raises the left operand to the power of the right operand.

Example:

```
# Division

result = 10 / 3  # Result: 3.3333...

# Modulus

remainder = 10 % 3  # Result: 1

# Floor Division

result = 10 // 3  # Result: 3

# Exponentiation

result = 2 ** 3  # Result: 8
```

Q4. Creating a list of length 10 with multiple types of data:

```
mixed_list = ['Hello', 123, 3.14, True, (1, 2), [4, 5], {'a': 1}, None, 'World', 10.5]

# Printing elements and their data types using a for loop
```

```
for element in mixed_list:
    print(element, type(element))
```

Q5. Using a while loop to verify if A is purely divisible by B:

```
A = 20
B = 4
count = 0
while A % B == 0:
    count += 1
    A /= B
Print ("A is divisible by B {} times".format(count))
```

Q6. Creating a list containing 25 int type data and printing if each element is divisible by 3 or not:

```
int_list = list(range(1, 26))
for num in int_list:
    if num % 3 == 0:
        print(num, "is divisible by 3")
    else:
        print(num, "is not divisible by 3")
```

Q7. Mutable and Immutable data types:

- **Mutable data types**: These are the data types whose values can be changed after they are created. Examples include lists, dictionaries, and sets.
- # Example of mutable data type
- mutable_list = [1, 2, 3]
- mutable_list[0] = 100
- print(mutable_list) # Output: [100, 2, 3]
-
- **Immutable data types**: These are the data types whose values cannot be changed after they are created. Examples include strings, tuples, and numbers.
- # Example of immutable data type
- immutable_tuple = (1, 2, 3)
- # immutable_tuple[0] = 100 # This will raise an error since tuples are immutable

The property of immutability ensures that the value of these data types remains constant throughout their lifetime. Any operation that seems to modify the value of an immutable object actually creates a new object with the modified value.