assignment =1

June 3, 2023

1 Q1. Create one variable containing following type of data:

(i) string
(ii) list
(iii) float
(iv) tuple

[3]:
```
var_string = "Hello"

var_list = ['jadavbhai',7,90.0,4+5j,[1,2,3]]

var_float = 9.0

var_tuple =('jadavbhai',9,8.0,6+7j,[1,2,3])
```

2 Q2. Given are some following variables containing data:

(i) var1 = ''
(ii) var2 ='[DS, ML .Python]'
(iii) var3 =['DS','ML','Python']
(iv) var4 =1.

What will be the data type of the above given variable

#(i) var1 = '' type(var1)

[6]:
```
#(i)
var1 = ''
type(var1)
```

[6]: str

[ ]:
```
#(ii)
var2 ='[ DS ,ML,Python ]'
type(var2)
```

[16]:
```
#(iii)
var3 = ['DS','ML','PYTHON']
type(var3)
```

[16]:
```
list
```

[5]:
```
#(iv)
var4 = 1.
type(var4)
```

[5]:
```
float
```

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

(i) /
(ii) %
(iii) //
(iv) **

[9]:
```
#(i) / : It gives you the divisible value of the 2 numbers.
div = 50/3
div
```

[9]:
```
16.666666666666668
```

[12]:
```
#(ii) % : It gives you the reminder value of the 2 numbers.
rem = 100%3
rem
```

[12]:
```
1
```

[18]:
```
#(i)// : It gives you the flour value of the 2 numbers. It like the integer␣
 ↪part of the divisble value
flour = 100//3

#Alternative
div_flour = int(100/3)

print(flour, div_flour)
```

```
33 33
```

[19]:
```
#(iv) **:It use for power value. second place number denotes the power of first␣
 ↪place number
```

```
pow = 7**9
pow
```

[19]: 40353607

4 4. Create a list of length 10 of your choice containing multiple types data . using for loop print the element and its data type.

[31]:
```
lst = ['sting',3,20.0,3+4j,[1,2,3],True,20,3.33334,0b100,bin(10)]

for i in lst:
    print (f'Element "{i}" type is  {type(i)}')
```

```
Element "sting" type is  <class 'str'>
Element "3" type is  <class 'int'>
Element "20.0" type is  <class 'float'>
Element "(3+4j)" type is  <class 'complex'>
Element "[1, 2, 3]" type is  <class 'list'>
Element "True" type is  <class 'bool'>
Element "20" type is  <class 'int'>
Element "3.33334" type is  <class 'float'>
Element "4" type is  <class 'int'>
Element "0b1010" type is  <class 'str'>
```

5 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 number A:"))
temp_A = A

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

count = 0
while temp_A%B==0:
    count+=1
    temp_A = temp_A/B


if count > 0:
    print(f'{A} is purely divisible by {B} in {count} number of times')
else:
    print(f'{A} is  not divisible by {B}')
```

```
Enter number A:32020
```

```
Enter number B:1
```

6 Q6.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.

[1]:
```
lst = range(25)

for i  in   lst:
    if i%3 == 0:
        print(f'{i} is divisible by 3')
    else:
        print(f'{i} is not divisible by 3 ')
```

```
0 is divisible by 3
1 is not divisible by 3
2 is not divisible by 3
3 is divisible by 3
4 is not divisible by 3
5 is not divisible by 3
6 is divisible by 3
7 is not divisible by 3
8 is not divisible by 3
9 is divisible by 3
10 is not divisible by 3
11 is not divisible by 3
12 is divisible by 3
13 is not divisible by 3
14 is not divisible by 3
15 is divisible by 3
16 is not divisible by 3
17 is not divisible by 3
18 is divisible by 3
19 is not divisible by 3
20 is not divisible by 3
21 is divisible by 3
22 is not divisible by 3
23 is not divisible by 3
24 is divisible by 3
```

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

ans. mutable data type like a list in which we remove , update , append or replace the index element on the bases of its index value .

[1]:
```
lst = ['string',3, 90.0, 3+4j , [1,2,3]]

lst[0] = 'hello'
lst. append("Append")
lst.remove(3)

lst
```

[1]:
```
['hello', 90.0, (3+4j), [1, 2, 3], 'Append']
```

but in immutable data type like string in which we can not remove , update , append or replace the index element on the bases of its index value.

[3]:
```
string = "Hello"

string = 'y'

string
```

[3]:
```
'y'
```

[ ]:
```

```