LIST

In [1]:
```
fruits = []
```

In [2]:
```
fruits.append("apple")
```

In [3]:
```
fruits
```

Out[3]:
```
['apple']
```

In [4]:
```
fruits.extend(['banana','cherry'])

fruits
```

Out[4]:
```
['apple', 'banana', 'cherry']
```

In [ ]:
```
fruits.remove('banana')
```

In [11]:
```
fruits
```

Out[11]:
```
['apple', 'cherry']
```

In [12]:
```
fruits.pop()
```

Out[12]:
```
'cherry'
```

In [13]:
```
fruits
```

Out[13]:
```
['apple']
```

In [14]:
```
fruits.append('banana')
```

In [15]:
```
fruits
```

Out[15]:
```
['apple', 'banana']
```

In [16]:
```
fruits.reverse()
```

In [17]:
```
fruits
```

Out[17]:
```
['banana', 'apple']
```

In [18]:
```
fruits.insert(1,'mango')


fruits
```

Out[18]:
```
['banana', 'mango', 'apple']
```

In [19]:
```
fruits.extend(['cherry','grapes'])

fruits
```

Out[19]: ['banana', 'mango', 'apple', 'cherry', 'grapes']

In [20]:
```
print(fruits[2:4])
```

```
['apple', 'cherry']
```

In [22]:
```
numb=[1,2,3,4,5]

numb
```

Out[22]: [1, 2, 3, 4, 5]

In [23]:
```
print(numb[0])
print(numb[-1])
```

```
1
5
```

In [25]:
```
numb[2]=10

numb
```

Out[25]: [1, 2, 10, 4, 5]

In [26]:
```
fruits
```

Out[26]: ['banana', 'mango', 'apple', 'cherry', 'grapes']

In [27]:
```
if "grape" in fruits:
    print("grape in the list")
else:
    print("grape is not in the list")
```

```
grape is not in the list
```

In [28]:
```
print("items in the list")
for item in fruits:
    print(item)
```

```
items in the list
banana
mango
apple
cherry
grapes
```

In [29]:
```
len(fruits)
```

Out[29]: 5

TUPLE

In [31]:
```
numbers=(1,2,3,4,5)

numbers.count(2)
```

Out[31]: 1

In [32]:
```
numbers.index(4)
```

Out[32]: 3

In [33]:
```
numbers_list = list(numbers)
numbers_list.append(5)
numbers = tuple(numbers_list)
print("Updated tuple after appending:", numbers)
```

```
Updated tuple after appending: (1, 2, 3, 4, 5, 5)
```

In [38]:
```
colors = ("red", "green", "blue")

colour=("red","green","blue")
colour[2]="black"
colour
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[38], line 4
      1 colors = ("red", "green", "blue")
      3 colour=("red","green","blue")
----> 4 colour[2]="black"
      5 colour

TypeError: 'tuple' object does not support item assignment
```

In [39]:
```
# Convert tuple into list
colour = list(colour)
# Append item in to list
colour.append("yellow")
# Convert list in to tuple
colour = tuple (colour)
colour
```

Out[39]: ('red', 'green', 'blue', 'yellow')

SET

In [40]:
```
my_set=set()
```

In [41]:
```
type(my_set)
```

Out[41]: set

In [42]:
```
my_set.add("apple")

my_set
```

Out[42]: {'apple'}

In [43]:
```
my_set.add("banana")

my_set
```

Out[43]: {'apple', 'banana'}

In [44]:
```
my_set.add("apple")
my_set
# apple is not added 2nd time
```

Out[44]: {'apple', 'banana'}

In [45]:
```
set1=[1,2,3]
set1=set(set1)
set2=[3,4,5]
set2=set(set2)
union_set = set1 | set2
union_set
```

Out[45]: {1, 2, 3, 4, 5}

In [46]:
```
intersection_set = set1 & set2
intersection_set
```

Out[46]: {3}

In [47]:
```
my_set.discard("apple")
my_set
```

Out[47]: {'banana'}

In [48]:
```
set3=[1,2,3,4,5,6,7,8,9]
set3=set(set3)
for i in set3:
    print(i)
```

```
1
2
3
4
5
6
7
8
9
```

DICTIONARY

In [49]:
```
my_dict={"name":"Ali"}
my_dict
```

Out[49]: {'name': 'Ali'}

In [50]:
```
my_dict["name"]
```

Out[50]: 'Ali'

In [51]:
```
my_dict={"name":"Ali"}
my_dict['age']=25
my_dict
```

Out[51]: {'name': 'Ali', 'age': 25}

In [52]:
```
my_dict.update({"name":"Ahmed"})
my_dict
```

Out[52]: {'name': 'Ahmed', 'age': 25}

In [53]:
```
student={"name": "Sara", "grade": "A", "age": 17}
student
```

Out[53]: {'name': 'Sara', 'grade': 'A', 'age': 17}

In [54]:
```
student.keys()
```

Out[54]: dict_keys(['name', 'grade', 'age'])

In [55]:
```
student.values()
```

Out[55]: dict_values(['Sara', 'A', 17])

In [56]:
```
student={"name": "Sara", "grade": "A", "age": 17}
check="grade"
if check in student:
    print("grade is exist")
else:
    print("grade is not exist")
```

```
grade is exist
```

In [57]:
```
student={"name": "Sara", "grade": "A", "age": 17}
for key,value in student.items():
    print(key,":",value)
```

```
name : Sara
grade : A
age : 17
```

IF-ELIF-ELSE

In [58]:
```
numbers = [5, 10, 15, 20, 25]

if(numbers[0]==10):
    print("Found 10")
elif(numbers[1]==10):
    print("Found 10")
elif(numbers[2]==10):
    print("Found 10")
elif(numbers[3]==10):
    print("Found 10")
elif(numbers[4]==10):
    print("Found 10")
else:
    print("10 not found")
```

```
Found 10
```

In [59]:
```
a=27
if a>20:
    print("Large Number")
else:
    print("Small Number")
```

```
Large Number
```

In [ ]:
```
# SIMPLE EXPRESSION
```

In [60]:
```
a=5
b=10
print(a+b)
print(a*b)
print(b-a)
print(b/a)
```

```
15
50
5
2.0
```

In [61]:
```
c=(a+b)*2
c
```

Out[61]: 30

In [62]:
```
age = int(input("Enter your age: "))

if age >= 18:
    print("You are an adult")
else:
    print("You are a minor")
```

```
you are not eligable
```

MORE EXPRESSION QUESTIONS

In [63]:
```
# area of a rectangle
width=5
height=10
area=width*height
area
```

Out[63]: 50

In [64]:
```
# converts Celsius to Fahrenheit:
C =25
F = (C * 9/5) + 32
F
```

Out[64]: 77.0

In [65]:
```
# simple interest

P = 1000
R = 5
```

```
T = 2

SI = (P*R*T)/100
SI
```

Out[65]: 100.0

RANGE FUNCTION

In [66]:
```
# simple interest

P = 1000
R = 5
T = 2

SI = (P*R*T)/100
SI
```

Out[66]: 100.0

In [67]:
```
# simple interest

P = 1000
R = 5
T = 2

SI = (P*R*T)/100
SI
```

Out[67]: 100.0

In [68]:
```
# simple interest

P = 1000
R = 5
T = 2

SI = (P*R*T)/100
SI
```

Out[68]: 100.0

In [69]:
```
# simple interest

P = 1000
R = 5
T = 2

SI = (P*R*T)/100
SI
```

Out[69]: 100.0

In [70]:
```
for i in range(10,0,-1):
    print(i)
```

```
10
9
8
7
6
5
4
3
2
1
```

In [71]:
```
list1=[]
for i in range(3,8):
    list1.append(i)
    print(list1)
```

```
[3]
[3, 4]
[3, 4, 5]
[3, 4, 5, 6]
[3, 4, 5, 6, 7]
```

In [72]:
```
for i in range(1,5):
    square = i**2
    print(square)
```

```
1
4
9
16
```

In [ ]: