Immutable Object Vs Mutable object

----

1) What is an immutable object ? ---> Can not be modified
   What is a mutable object ? ---> Can be modified
2) What are the 5 + 3 = 8 immutable objects ? ---> int , float , complex , bool , NoneType
                                                                                                    and
                  str , tuple , range
    What are the three mutable objects ? ---> List , set and dict
3) Is it possible to add / remove elements to / from immutable object ? --->
          No becoz they are neither growable nor shrinkable
   What about mutable object ? ---> Elements can be appended and removed from mutable object
4) Are immutable objects reusable ? ---> Yes except range object
   What about mutable objects ? ---> They are not reusable
5) What is reusability ? ---> If same object already exists in memory,
                                             it will be reused and new object is not created
6) What happens when an object does not exist ? ---> A new object will be created
7) a = 25
   b = 25
   What does b = 25 do ? ---> Reference 'b' points to same object 25 becoz 25 is immutable and reusable
   What is the result of a is b ? ---> True becoz references 'a' and 'b' point to same object 25
8) a = [10 , 20 , 30]
   b = [10 , 20 , 30]
   What does b = [10 , 20 , 30] do ? --->
      Refence 'b' points to new list even though [10 , 20 , 30] already exists in memory
            becoz list is mutable and not reusable
   What is the result of a is b ? ---> False becoz references 'a' and 'b' point to different lists
9) a = range(5)
   b = range(5)
   What does b = range(5) do ? ---> Refence 'b' points to new range object even though range object already
          exists in memory becoz range is immutable but not reusable
10) a = 25
     a = 35
    What is modified when a = 35 is executed ? ---> Reference but not object
11) In other words, reference 'a' points to another object 35
12) Why is object not modified when a = 35 is executed ? ---> Since int object is immutable
                         and hence it can not be modified

```
# Find outputs
a = 25 # Ref 'a' points to object 25
print(id(a)) # Address of object 25 (may be 1000)
a = 35 # Ref 'a' is modified to another object 35
print(id(a)) # Address of object 35 (may be 2000)
'''
```

```
1) a = 25
   a = 35
   Why is 25 not replaced with 35 when a = 35 is executed ? ---> Since int object is immutable
2) What is modified when a = 35 is executed ? ---> Reference but not object
3) How many objects are in the program ? ---> Two i.e. 25 and 35
'''

# Find outputs (Home work)
a = 25.7 # Ref 'a' points float object 25.7
print(id(a)) # Address of object 25.7 (1000)
print(a) # 25.7
a = 35.6 # Ref 'a' is modified to another object 35.6
print(id(a)) # Address of object 35.6 (2000)
print(a) # 35.6
b = True # Ref 'b' points object True
print(id(b)) # Address of object True (3000)
b = False # Ref 'b' is modified to another object False
print(id(b)) # Address of object False (4000)
c = None # Ref 'c' points object None
print(id(c)) # Address of object None (5000)
c = None # Nothing is modified
print(id(c)) # Same address (5000)


# Find outputs (Home work)
a = 'Hyd' # Ref 'a' points to 'Hyd'
print(id(a)) # Address of object 'Hyd' (may be 1000)
#a[1] = 'e' # Error becoz str object is immutable
a = 'Sec' #Ref 'a' is modified to another object 'Sec'
print(id(a)) # Address of object 'Sec' (may be 2000)
b = (10 , 20 , 15 , 18) # Ref 'b' points to tuple
print(id(b)) # Address of tuple (may be 3000)
#b[2] = 19 # Error becoz tuple is immutable
b = (30 , 40 , 35 , 32) #Ref 'b' is modified to another tuple
print(id(b)) # Address of 2nd tuple (may be 4000)
c = range(5) # Ref 'c' points to range object
print(id(c)) # Address of range object (may be 5000)
#c[3] = 10 # Error becoz range object is immutable
c = range(5) #Ref 'c' is modified to another range object becoz it is not reusable
print(id(c)) # Address of 2nd range object (may be 6000)
'''
1) a = 'Hyd'
   a = 'Sec'
   What is modified when a = 'Sec' is executed ? ---> Reference but not object
2) b = (10 , 20 , 15 , 18)
```

```
   b = (30 , 40 , 35 , 32)
   What is modified when b = (30 , 40 , 35 , 32) is executed ? ---> Reference but not object
3) c = range(5)
   c = range(5)
   What is modified when c = range(5) is executed ? ---> Reference but not object
'''

# Find outputs (Home work)
a = 25 # Ref 'a' points to object 25
b = 25 # Ref 'b' points to same object 25 becoz int is immutable and reusable
print(a is b) # True becoz 'a' and 'b' point to same object 25
c = 'Hyd' # Ref 'c' points to object 'Hyd'
d = 'Hyd' # Ref 'd' points to same object 'Hyd' becoz str is immutable and reusable
print(c is d) # True becoz 'c' and 'd' point to same object 'Hyd'
e = False # Ref 'e' points to object False
f = False # Ref 'f' points to same object False becoz bool is immutable and reusable
print(e is f) # True becoz 'e' and 'f' point to same object False
g = range(10) # Ref 'g' points to range object
h = range(10) # Ref 'h' points to another range object becoz range is immutable but not reusable
print(g is h) # False becoz 'g' and 'h' point to different range objects
'''
1) Can there be multiple int objects with same value ? ---> No becoz int object is immutable and reusable
2) Can there be multiple float objects with same value ? ---> No becoz float object is immutable and reusable
3) Can there be multiple string objects with same string ? ---> No becoz str object is immutable and reusable
4) Can there be multiple lists with same elements ? ---> Yes becoz list is mutable and not reusable
5) Can there be multiple tuples with same elements ? ---> No becoz tuple is immutable and reusable
6) Can there be multiple sets with same elements ? ---> Yes becoz set is mutable and not reusable
7) Can there be multiple dictionaries with same key : value pairs ? ---> Yes becoz dict is mutable and not reusable
8) Which objects are reusable (5 + 2 = 7) ? ---> int , float , bool , complex , NoneType , str and tuple
9) Which objects are not reusable (1 + 3 = 4) ? ---> range , list , set , dict , bytearray
10) What does 'is' operator do ? ---> Compares refernces but not objects
    What does == operator do ? ---> Compares objects
'''

 #Find outputs(Home work)
a = [10 , 20 , 15 , 18] # Ref 'a' points to list
b = [10 , 20 , 15 , 18] # Ref 'b' points to another list becoz list is mutable and not reusable
print(a is b) # False becoz 'a' and 'b' point to different lists
c = {10 : 20, 30 : 40} # Ref 'c' points to dictionary
d = {10 : 20, 30 : 40} # Ref 'd' points to another dictionary becoz dict is mutable and not reusable
print(c is d) # False becoz 'c' and 'd' point to different dictionaries
e = (10 , 20 , 15 , 18) # Ref 'e' points to tuple
f = (10 , 20 , 15 , 18)# Ref 'f' points to same tuple becoz tuple is immutable and reusable
print(e is f) # True becoz 'e' and 'f' point to same tuple
```

```
g = {10 , 20 , 15 , 18} # Ref 'g' points to set
h = {10 , 20 , 15 , 18} # Ref 'h' points to another set becoz set is mutable and not reusable
print(g is h) # False becoz 'g' and 'h' point to different sets
```