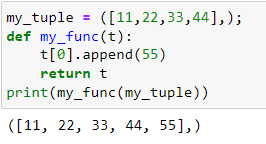
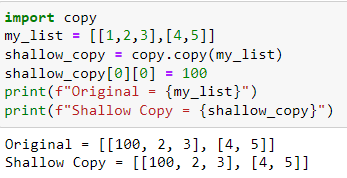
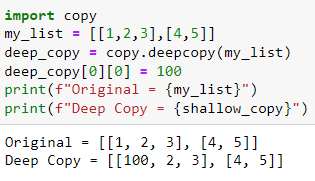


When you pass the list to the function,it means that you are passing the reference of the list that is the actual memory where it is present and not the copy that’swhy you observe the change in the original list outside the function

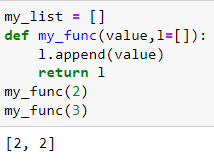


We can say that tuples are immutable but if they contain list then we can modify them as lists are mutable but when you try to append any element at first index,you will face an error since append id th method of list not of tuple.In ddition if you want to assign the value for example 55 to zero index it will throw an error since tuple object does not support item assignment





Shallow copy creates a new object but shares references to nested elements in the list while deep copy creates an entirely new object and has all new references



when the function is defined the defult parameters gets evaluated that time only when and upon calling the function you just use them so when 2 was passed the empty list appends 2 in it and become a list of one number hat is 2 but when you again called the function 3 got appended into the list that already contained 2

class Book\_data:

def \_\_init\_\_(self,\_id,title,author,year,borrowed):

self.\_id = \_id

self.title = title

self.author = author

self.year = year

self.borrowed = borrowed

class Book\_management:

def \_\_init\_\_(self):

self.books = []

def add\_book(self,book):

self.books.append({"ID":book.\_id,"Title":book.title,"Author":book.author,"Year":book.year,"Borrowed":book.borrowed})

print(f"Book \*{book.title}\* Added Successfully")

def list\_book(self):

print(self.books)

def borrow\_book(self):

ans = input("Enter the name of the book you want to borrow:")

for book in self.books:

if ans.lower() in book["Title"].lower():

if book["Borrowed"]:

print("Book Already issued to someone else")

break;

if not book["Borrowed"]:

book["Borrowed"] = True

print(book["Title"],"Issued")

def return\_book(self):

ans = input("Enter the name of the book you want to borrow:")

for book in self.books:

if ans.lower() in book["Title"].lower():

if book["Borrowed"]:

print("Returned Successfully!")

book["Borrowed"] = False

break;

if not book["Borrowed"]:

print("This book is not borrowed Man!!!")

book\_manager = Book\_management()

book\_1 = Book\_data(1,"Software Construction","ABC",2023,False)

book\_2 = Book\_data(1,"Embedded Systems","XYZ",2022,False)

book\_manager.add\_book(book\_1)

book\_manager.add\_book(book\_2)

while(True):

ans = int(input("Select \n(1)Add \n(2)View \n(3)Borrow \n(4)Return \n(5)Search\n"))

if ans == 1:

book\_1 = Book\_data(1,"Software Construction","ABC",2023,False)

book\_2 = Book\_data(2,"Embedded Systems","XYZ",2022,True)

book\_manager.add\_book(book\_1)

book\_manager.add\_book(book\_2)

elif ans == 2:

book\_manager.list\_book()

elif ans == 3:

book\_manager.borrow\_book()

elif ans == 4:

book\_manager.return\_book()

asked = input("Do you want to continue (y/n)?").lower()

if asked != "y":

break;