```python
# 1. Library Management System
#    - Create a system to manage books in a library.
#      - Tasks:
#      1. Store book details like title, author, and availability using appropriate data types.
#      2. Allow the user to borrow or return books. Use conditional statements to check availability.
#      3. Implement loops to display all available books.
#      4. Handle exceptions for invalid inputs (e.g., trying to borrow a non-existent book).
#      5. Use a module to manage library operations (e.g., a separate module for adding/removing books).
import Remove
import Add
#      1. Store book details like title, author, and availability using appropriate data types
books = {
   "The Great Gatsby": {"author": "F. Scott Fitzgerald", "availability": True},
   "1984": {"author": "George Orwell", "availability": False},
   "To Kill a Mockingbird": {"author": "Harper Lee", "availability": True},
   "Pride and Prejudice": {"author": "Jane Austen", "availability": True},
   "Moby-Dick": {"author": "Herman Melville", "availability": False},
   "The Catcher in the Rye": {"author": "J.D. Salinger", "availability": True},
   "The Odyssey": {"author": "Homer", "availability": True},
   "War and Peace": {"author": "Leo Tolstoy", "availability": False},
   "The Hobbit": {"author": "J.R.R. Tolkien", "availability": True},
   "Crime and Punishment": {"author": "Fyodor Dostoevsky", "availability": False}
}
#      3. Implement loops to display all available books.
def Display():
    global books
    for k,v in books.items():
       if v['availability']==True:
          print(k,"Author:",v['author'])

print("1.Borrow Book")
print("2.return Book")
print("3.Display Book List")
print("4.Exit")
n=int(input("enter your choice "))
while(n!=4):
   if n==1:
      Remove.borrow(books)
   elif n==2:
      Add.Return(books)
```

```
  elif n==3:
    Display()
  if n==4:
     break
  n=int(input("if you want to continue press Choice "))
  print("1.Borrow Book")
  print("2.return Book")
  print("3.Exit")
```

```
#     2. Allow the user to borrow or return books. Use conditional statements to check availability.
```

Remove.py

```
def borrow(d):
    b=input("Enter book name ")
    if b in d.keys():
        if d[b]['availability']==True:
            print("Please take your book")
            d.pop(b)
        else:
            print("You can't borrow this Book is currently not available")
    else:
        print("The book is not in book list")
```

Add.py

```
def Return(d):
  try:
    s=input("Enter book name ")
    author=input("Enter Author name ")
    d.update({s:{'author':author,'availability':True}})
    print("Book Returned Succesfully")
  except:
    print("Please Enter Correctly")
```

output:

Borrow

```
Modules_Assignments/Module_Assignment.py
1.Borrow Book
2.return Book
3.Display Book List
4.Exit
enter your choice 1
Enter book name 1984
You can't borrow this Book is currently not available
if you want to continue press Choice 1
1.Borrow Book
2.return Book
3.Exit
Enter book name the habbit
The book is not in book list
if you want to continue press Choice 1
1.Borrow Book
2.return Book
3.Exit
Enter book name The Hobbit
Please take your book
if you want to continue press Choice 3
1.Borrow Book
2.return Book
3.Exit
The Great Gatsby Author: F. Scott Fitzgerald
To Kill a Mockingbird Author: Harper Lee
Pride and Prejudice Author: Jane Austen
The Catcher in the Rye Author: J.D. Salinger
The Odyssey Author: Homer
if you want to continue press Choice
```

Return :

```
if you want to continue press Choice 2
1.Borrow Book
2.return Book
3.Exit
Enter book name The Hobbit
Enter Author name R R Darvin
Book Returned Succesfully
if you want to continue press Choice 3
1.Borrow Book
2.return Book
3.Exit
The Great Gatsby Author: F. Scott Fitzgerald
To Kill a Mockingbird Author: Harper Lee
Pride and Prejudice Author: Jane Austen
The Catcher in the Rye Author: J.D. Salinger
The Odyssey Author: Homer
The Hobbit Author: R R Darvin
if you want to continue press Choice
```

Exit :

```
if you want to continue press Choice 4
PS C:\Lavanya_Code\Python_Lectures_Assignments>
```

```
#  2. Restaurant Billing System

#   - Develop a billing system for a restaurant.
#   - Tasks:
#     1. Create a menu using a dictionary where keys are item names and values are prices.
#     2. Use a loop to allow customers to order multiple items.
#     3. Calculate the total bill and apply a conditional discount if the bill exceeds a certain
amount.
#     4. Handle exceptions for invalid item selection.
#     5. Organize the code into modules for the menu, order processing, and billing.

#     1. Create a menu using a dictionary where keys are item names and values are prices.
'''
import Menu
import Order_procesing
import Billing
Menu.DiplayMenu()
Total_Bill=Order_procesing.Order(Menu.menu)
Bill=Billing.Bill(Total_Bill)
print(Bill)
'''
```

Billing –Module-Billing.py

```
#     3. Calculate the total bill and apply a conditional discount if the bill exceeds a certain
amount.
def Bill(Total_Bill):
   if Total_Bill>200 and Total_Bill<1000:
      print("Congrasulations You Got Discount of 20% on above 200 Bill")
      return f'Total Bill: {Total_Bill} Discounted Bill: {Total_Bill-(20/100*Total_Bill)}'
   elif Total_Bill>1000:
      print("Congrasulations You Got Discount of 50% on above 1000 Bill")
      return f'Total Bill: {Total_Bill} Discounted Bill: {Total_Bill-(50/100*Total_Bill)}'
   else:
      return f'Your Total Bill {Total_Bill}'
```

Order-Processing:

```
def Order(menu):

 Total_Bill=0
 print("Please Enter Your Items once done enter done")
 #     2. Use a loop to allow customers to order multiple items.
 while(True):
  s=input("please Enter name of Item ")
  if s=='Done':
    break
  else:
   if s in menu:
     Total_Bill+=menu[s]
   else:
    print("Please Enter Items of Menu only")
 return Total_Bill
```

Output:

```
Modules_Assignments/Module_Assignment.py
Burger 90
Pizza 170
Pasta 50
Salad 50
Tacos 99
Soup 70
Sandwich 30
Fries 80
Steak 120
Ice Cream 75
Please Enter Your Items once done enter done
please Enter name of Item Pizza
please Enter name of Item Ice Cream
please Enter name of Item Done
Congrasulations You Got Discount of 20% on above 200 Bill
Total Bill: 245 Discounted Bill: 196.0
PS C:\Lavanya_Code\Python_Lectures_Assignments>
```

```
Ice Cream 75
Please Enter Your Items once done enter done
please Enter name of Item Ice Cream
please Enter name of Item Pizza
please Enter name of Item Pizza
please Enter name of Item Pizza
please Enter name of Item Pizza
please Enter name of Item Pizza
please Enter name of Item Burger
please Enter name of Item Steak
please Enter name of Item Done
Congrasulations You Got Discount of 50% on above 1000 Bill
Total Bill: 1135 Discounted Bill: 567.5
PS C:\Lavanya_Code\Python_Lectures_Assignments>
```

```
Ice Cream 75
Please Enter Your Items once done enter done
please Enter name of Item Soup
please Enter name of Item Salad
please Enter name of Item Done
Your Total Bill 120
PS C:\Lavanya_Code\Python_Lectures_Assignments>
```

Items not in menu

```
Ice Cream 75
Please Enter Your Items once done enter done
please Enter name of Item cock
Please Enter Items of Menu only
please Enter name of Item
```

```
# 3. Student Grading System
#    - Build a system to calculate grades for students.
#    - Tasks:
#      1. Accept student details and marks for subjects using appropriate data structures.
#      2. Use conditions to assign grades based on marks (e.g., A, B, C).
#      3. Use loops to calculate the average marks of multiple students.
#      4. Handle exceptions for invalid marks (e.g., marks outside the 0-100 range).
#      5. Create a module to format and display the student results.
import Result
students={
   'Alice': 76,
   'Bob': 45,
   'Charlie': 92,
   'David': 58,
   'Eva': 30,
   'Fay': 84,
   'George': 71,
   'Helen': 99,
   'Ivy': 500,
   'Jack': 64
}

name=input('Please Enter name of Student you want to see result ')
import Result
Result.result(name,students[name])

sum=0
l=0
for v in students.values():
   if v<=100:
      sum+=v
      l+=1

#      3. Use loops to calculate the average marks of multiple students.
print("Avrage Result Of Class is",sum/l)
```

Result:

```python
def result(name,marks):
 if marks>100:
   print(name,"You Got Invalid Result ")
 else:
   print(f'-------------{name}-----------------')
   if marks>=75 and marks<=100:
      print("Total Marks:",marks,"Grade:",'A')
   elif marks>=55 and marks<=74:
      print("Total Marks:",marks,"Grade:",'B')
   elif marks>=35 and marks<=54:
      print("Total Marks:",marks,"Grade:",'C')
   elif marks<35:
      print("Your Failed ")
```

Output:

```
Modules_Assignments/Module_Assignment.py
Please Enter name of Student you want to see result Ivy
Ivy You Got Invalid Result
Avrage Result Of Class is 72.44444444444444
PS C:\Lavanya_Code\Python_Lectures_Assignments> & C:/Users/Admin/AppData/Loc
Modules_Assignments/Module_Assignment.py
Please Enter name of Student you want to see result Helen
-------------Helen-----------------
Total Marks: 99 Grade: A
Avrage Result Of Class is 72.44444444444444
PS C:\Lavanya_Code\Python_Lectures_Assignments> & C:/Users/Admin/AppData/Loc
Modules_Assignments/Module_Assignment.py
Please Enter name of Student you want to see result Eva
-------------Eva-----------------
Your Failed
Avrage Result Of Class is 68.77777777777777
PS C:\Lavanya_Code\Python_Lectures_Assignments>
```

4. ATM Simulation System

```
#    - Simulate the operations of an ATM.
#    - Tasks:
#      1. Store account details (e.g., account number, balance) using a dictionary.
#      2. Implement withdrawal, deposit, and balance check functionalities.
#      3. Use conditional statements to check for sufficient balance before withdrawal.
#      4. Handle exceptions for invalid transactions (e.g., entering a non-numeric value for
withdrawal).
#      5. Create separate modules for authentication and transaction processing.

import authentication
import Withdraw
#      1. Store account details (e.g., account number, balance) using a dictionary.
accounts = {
    1001000: 1500.75,
    1011123: 3200.50,
    1089456: 450.30,
    1004: 8900.00,
    1090: 1234.67,
    1006: 345.10,
    10075656: 7600.99,
    1088: 234.55,

    100967676: 1750.80,
    10105656: 2895.25
}
def Deposit(acc):
    global accounts
    balance=int(input("Enter amount "))
    accounts[acc]=accounts[acc]+balance
    print('Deposit succefull')

def CheckBalance(acc):
    global accounts
    print("Your Balance:",accounts[acc])
acc=authentication.isauthor(input("Enter Account number"),accounts)
print(acc)
```

```
#     2. Implement withdrawal, deposit, and balance check functionalities.
if acc!='No':
   print("Please choose you Choice")
   print("1.Deposit")
   print("2.Withdraw")
   print("3.CheckBalance")
   print("4.Exist")
   n=int(input("Enter Your Chioce "))
   while(n!=4):
       if n==1:
            Deposit(acc)
       elif n==2:
         Withdraw.withdraw(acc,accounts)
       elif n==3:
           CheckBalance(acc)
       print('if you want continue Enter choice else enter 4')
       print("1.Deposit")
       print("2.Withdraw")
       print("3.CheckBalance")
       n=int(input("enter choice "))


       if n==4:
           break

else:
    print("Try After 24 Hr ")
```

Authentication-Model .py

```
#     5. Create separate modules for authentication and transaction processing.

def isauthor(acc,d):
 i=0
 while(i<2):
    try:
        if int(acc) in d:
            print("Loggined Succefully")
            return int(acc)
        else:
           print("Account number is Invalid")
           i+=1
           acc=input("Enter Account Number ")
           if i==2:
              if int(acc) in d:
                 print("Loggined Succefully")
                 return int(acc)


    except:
       acc=input("Please enter Account number in positive integer")
 return "No"
```

Withdrawal Model .py

```
#     3. Use conditional statements to check for sufficient balance before withdrawal.
#     4. Handle exceptions for invalid transactions (e.g., entering a non-numeric value for withdrawal).
def withdraw(acc,d):

    while(True):

     try:
       amount=int(input("enter amount "))
       if d[acc]>=amount:
         d[acc]=d[acc]-amount
         print("Withdrawal Sussefully")
         return
       else:
         print("Insuffiensint Balance ")
         return


     except:
       print("Enter amount in positive integer ")
```

Output:

Authentication test

Fails after 3 attempt

```
Modules_Assignments/Module_Assignment.py
Enter Account number9
Account number is Invalid
Enter Account Number 100
Account number is Invalid
Enter Account Number 1008
Try After 24 Hr
PS C:\Lavanya_Code\Python_Lectures_Assignments>
```

If 3rd attempt is right

```
Modules_Assignments/Module_Assignment.py
Enter Account number100
Account number is Invalid
Enter Account Number 100
Account number is Invalid
Enter Account Number 1006
Loggined Succefully
Please choose you Choice
1.Deposit
2.Withdraw
3.CheckBalance
4.Exist
Enter Your Chioce
```

If invalid input

```
PS C:\Lavanya_Code\Python_Lectures_Assignments> & C:/Users/Admin/A
Modules_Assignments/Module_Assignment.py
Enter Account numberlava
Please enter Account number in positive integer9.0
Please enter Account number in positive integer9.0
Please enter Account number in positive integer9.0
Please enter Account number in positive integer1006
Loggined Succefully
Please choose you Choice
1.Deposit
2.Withdraw
3.CheckBalance
4.Exist
Enter Your Chioce
```

Deposite Test:

```
Enter Account number1006
Loggined Succefully
1006
Please choose you Choice
1.Deposit
2.Withdraw
3.CheckBalance
4.Exist
Enter Your Chioce 1
Enter amount 100
Deposit succefull
if you want continue Enter choice
```

Check balance

```
Loggined Succefully
1006
Please choose you Choice
1.Deposit
2.Withdraw
3.CheckBalance
4.Exist
Enter Your Chioce 3
Your Balance: 345.1
if you want continue Enter choice else enter 4
```

```
enter choice 1
Enter amount 100
Deposit succefull
if you want continue Enter choice else enter 4
1.Deposit
2.Withdraw
3.CheckBalance
enter choice 3
Your Balance: 445.1
if you want continue Enter choice else enter 4
1.Deposit
```

Withdrawal:

```
Enter Account number1006
Loggined Succefully
1006
Please choose you Choice
1.Deposit
2.Withdraw
3.CheckBalance
4.Exist
Enter Your Chioce 3
Your Balance: 345.1
if you want continue Enter choice else enter 4
1.Deposit
2.Withdraw
3.CheckBalance
enter choice 2
enter amount 4000
Insuffiensint Balance
if you want continue Enter choice else enter 4
1.Deposit
2.Withdraw
3.CheckBalance
enter choice 2
enter amount 200
Withdrawal Sussefully
if you want continue Enter choice else enter 4
1.Deposit
2.Withdraw
3.CheckBalance
enter choice 3
Your Balance: 145.10000000000002
if you want continue Enter choice else enter 4
1.Deposit
2.Withdraw
3.CheckBalance
enter choice
```

Withdrawal of exceptions:

```
Enter Your Chioce 2
enter amount lo
Enter amount in positive integer
enter amount 9.0
Enter amount in positive integer
enter amount 100
Withdrawal Sussefully
if you want continue Enter choice else enter 4
1.Deposit
2.Withdraw
3.CheckBalance
enter choice
```

5. Inventory Management System

```
#   - Create a system to manage product inventory.
#   - Tasks:
#     1. Store product details like name, price, and stock quantity using a list of dictionaries.
#     2. Allow the user to add or remove products from the inventory using conditions.
#     3. Use loops to display all products and their details.
#     4. Handle exceptions for invalid inputs (e.g., negative stock quantity).
#     5. Organize the inventory operations in a module for better structure.

import AddItem
import RemoveItem
l=[{'name': 'Laptop', 'price': 1200.99, 'stock': 30},
{'name': 'Smartphone', 'price': 899.49, 'stock': 50},
{'name': 'Headphones', 'price': 199.99, 'stock': 75},
{'name': 'Smartwatch', 'price': 299.99, 'stock': 40},
{'name': 'Tablet', 'price': 350.50, 'stock': 60},
{'name': 'Keyboard', 'price': 45.99, 'stock': 100},
{'name': 'Mouse', 'price': 25.75, 'stock': 120},
{'name': 'Monitor', 'price': 350.00, 'stock': 25},
{'name': 'External Hard Drive', 'price': 85.49, 'stock': 40},
{'name': 'Wireless Charger', 'price': 35.00, 'stock': 150}]

print('''
   1.Add
   2.Remove
   3.Display
   4.Exit
''')
ch=int(input("Enter choice "))
while(ch!=4):
   if ch==1:
      AddItem.add(l)
   elif ch==2:
      RemoveItem.remove(l)
   elif ch==3:
      for i in l:
         for k,v in i.items():
            print(k,'->',v,"  ",end=" ")
         print()
   ch=int(input("Enter your choice "))
```

AddItems Model- add(Item).py

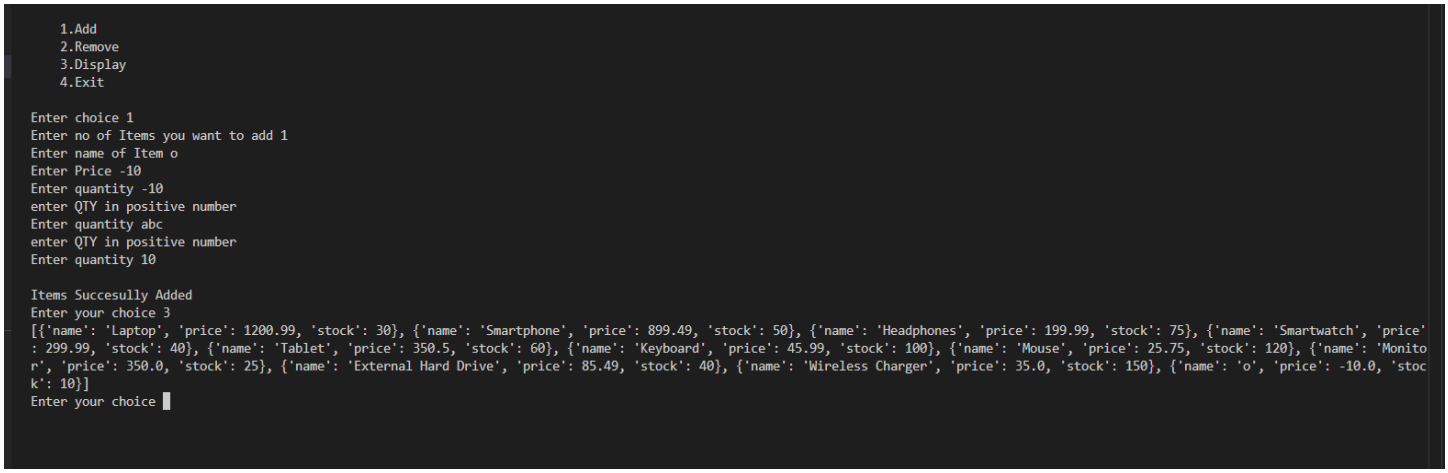
```
def add(Item):
    for i in range(int(input("Enter no of Items you want to add "))):
      name=input("Enter name of Item ")
      price=float(input("Enter Price "))
      while(True):
        try:

          QTY=int(input("Enter quantity "))
          if QTY<=0:
            print("enter QTY in positive number")
          else:
            Item.append({'name':name,'price':price,'stock':QTY})
            print("")
            break
        except:
          print("enter QTY in positive number")
    print("Items Succesully Added")
```

RemoveItems Model- add(Item).py

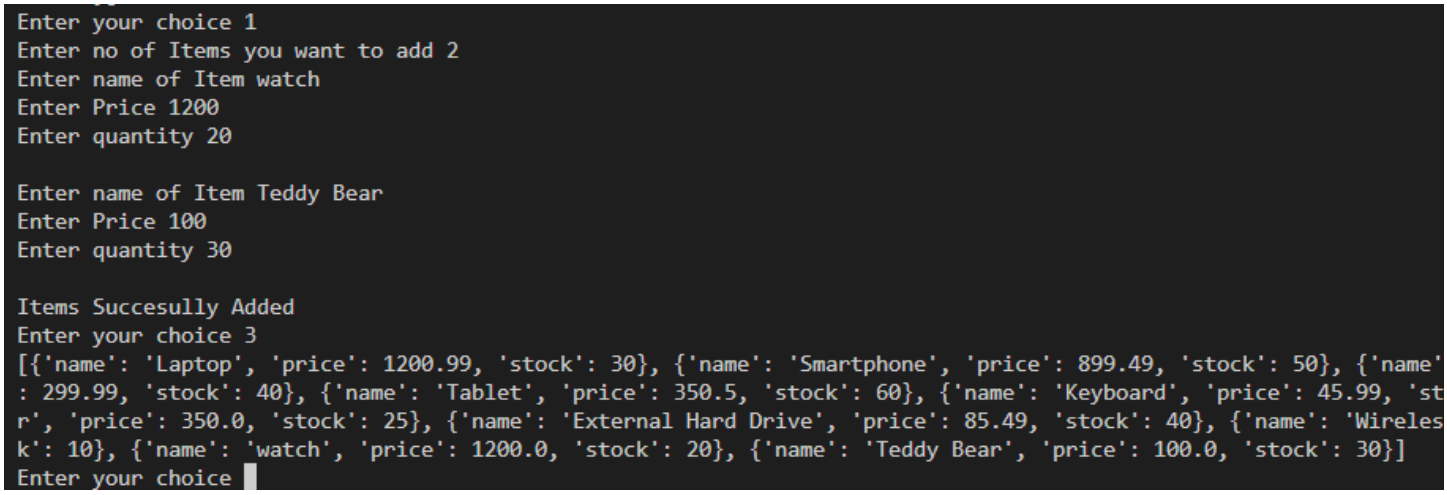
```
def remove(Items):
    print(Items)
    i=0
    n=(int(input("Enter number of Items you want to remove ")))
    while(i<n):
        name=input("Enter name ")
        f=1
        for d in Items:
           if name in d.values():
               f=0
               QTY=int(input("Enter Quntity "))
               i+=1
               if QTY>d['stock']:
                   print("Insuffient Quntity ")
                   break
               else:
                   d['stock']-=QTY
                   break
        if f==1:
          print("Item Not in List")
```

Output:

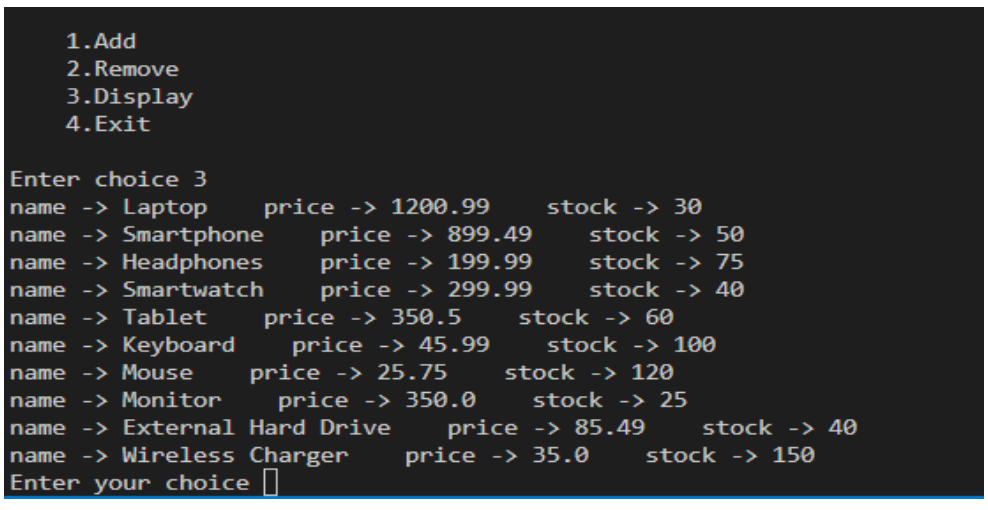
Adding QTY as string or negative number



Adding in right way



Display Items



Remove Items:

If Item not found

```
Enter choice 2
[{'name': 'Laptop', 'price': 1200.99, 'stock': 30}, {'name': 'Smartphone', 'price': 899.49, 'stock': 50}, {'name': 'Headphones', 'price': 199.99, 'stock': 75}, {'name': 'Smartwatch', 'price'
: 299.99, 'stock': 40}, {'name': 'Tablet', 'price': 350.5, 'stock': 60}, {'name': 'Keyboard', 'price': 45.99, 'stock': 100}, {'name': 'Mouse', 'price': 25.75, 'stock': 120}, {'name': 'Monito
r', 'price': 350.0, 'stock': 25}, {'name': 'External Hard Drive', 'price': 85.49, 'stock': 40}, {'name': 'Wireless Charger', 'price': 35.0, 'stock': 150}]
Enter number of Items you want to remove 2
Enter name po
Item Not in List
Enter name po
Item Not in List
Enter name po
Item Not in List
Enter name Mouse
Enter Quntity 20
Enter name po
Item Not in List
Enter name Monitor
Enter Quntity 20
Item Not in List
Enter your choice 3
name -> Laptop    price -> 1200.99    stock -> 30
name -> Smartphone    price -> 899.49    stock -> 50
name -> Headphones    price -> 199.99    stock -> 75
name -> Smartwatch    price -> 299.99    stock -> 40
name -> Tablet    price -> 350.5    stock -> 60
name -> Keyboard    price -> 45.99    stock -> 100
name -> Mouse    price -> 25.75    stock -> 100
name -> Monitor    price -> 350.0    stock -> 5
name -> External Hard Drive    price -> 85.49    stock -> 40
name -> Wireless Charger    price -> 35.0    stock -> 150
Enter your choice
```

If QTY is greater

```
    1.Add
    2.Remove
    3.Display
    4.Exit

Enter choice 2
[{'name': 'Laptop', 'price': 1200.99, 'stock': 30}, {'name': 'Smartphone', 'price': 899.49, 'stock': 50}, {'name': 'Headphones', 'price': 199.99, 'stock': 75}, {'name': 'Smartwatch', 'price'
: 299.99, 'stock': 40}, {'name': 'Tablet', 'price': 350.5, 'stock': 60}, {'name': 'Keyboard', 'price': 45.99, 'stock': 100}, {'name': 'Mouse', 'price': 25.75, 'stock': 120}, {'name': 'Monito
r', 'price': 350.0, 'stock': 25}, {'name': 'External Hard Drive', 'price': 85.49, 'stock': 40}, {'name': 'Wireless Charger', 'price': 35.0, 'stock': 150}]
Enter number of Items you want to remove 1
Enter name Mouse
Enter Quntity 200
Insuffient Quntity
Enter your choice
```