Chapter-10 Data Structure - II

Stack and Queue

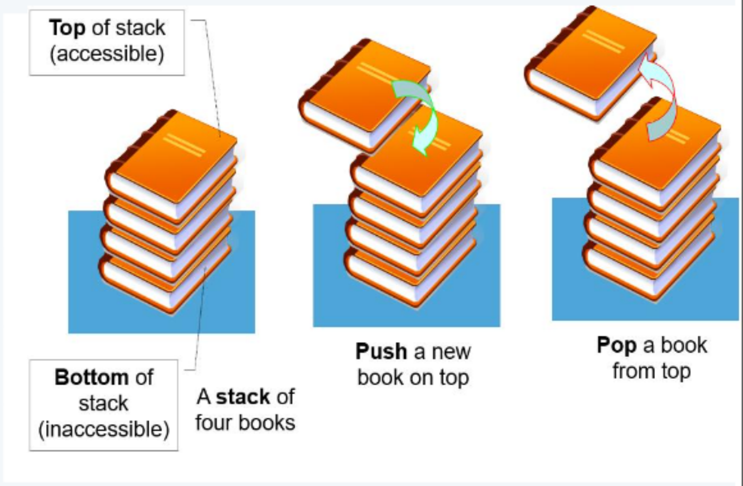
In this tutorial we will discuss the following topics

Chapter-10 Data Structure - II

Linear data structures are collections of components arranged in a straight line

1) Stack:

- ➔ A stack is a data structure that keeps objects in **Last-In-First-Out (LIFO)** order
- ➔ Objects are added to the top of the stack
- ➔ Only the top of the stack can be accessed
- ➔ A pile of books or a stack of dinner plates can be thought of examples of stacks.


Top of stack (accessible)
Bottom of stack (inaccessible)
A **stack** of four books
Push a new book on top
Pop a book from top

It has three primitive operations:

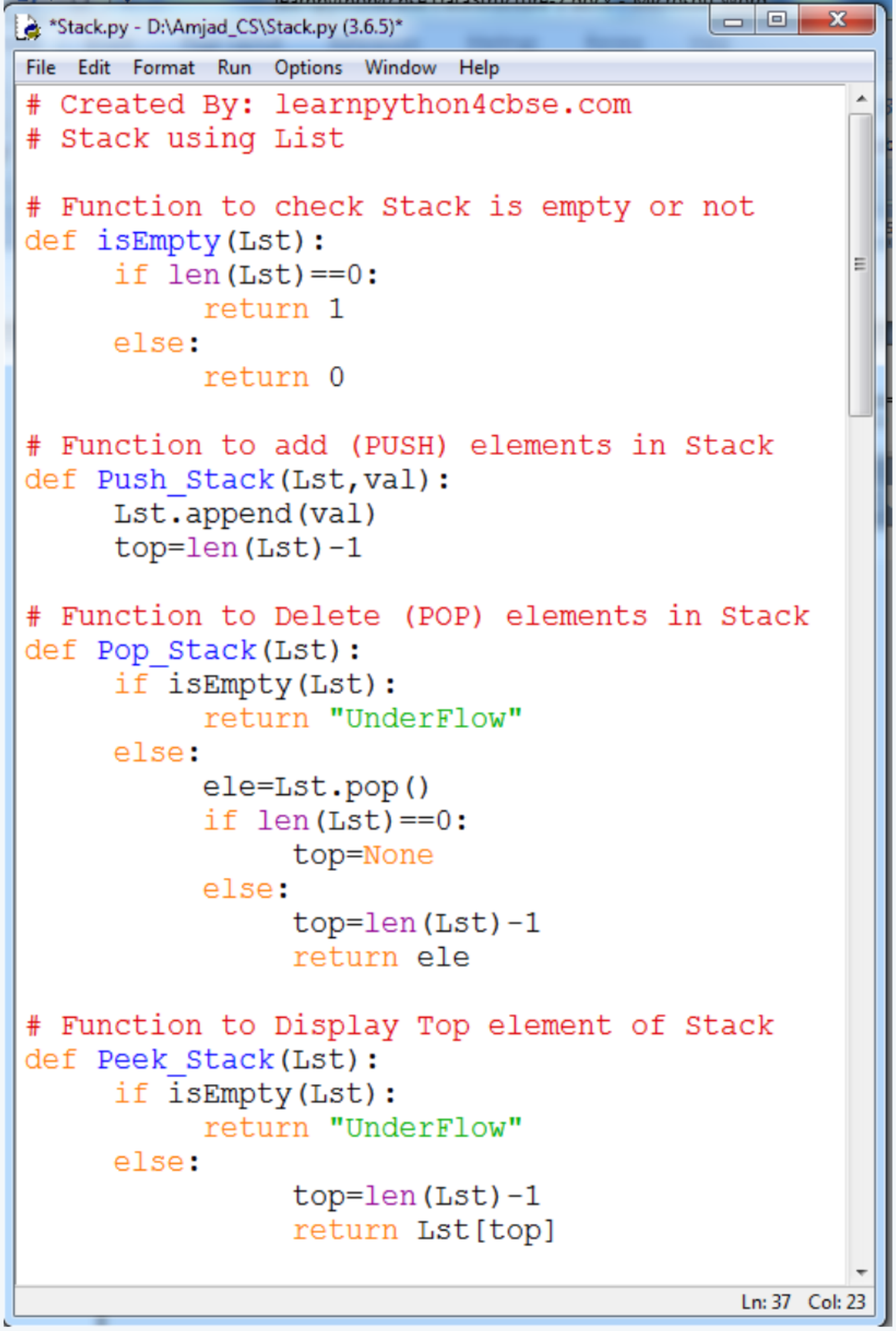
- **Push:** Add an element to the stack
- **Pop:** Remove an element from the stack
- **Peek:** Get the topmost element of the stack

In Python, a stack is implemented using a list object.

- To push an item in the stack, use the list function *append* list.append(item)
- To pop an item in the stack, use the list function *pop* list.pop()
- To get the top most item in the stack, write list[-1]

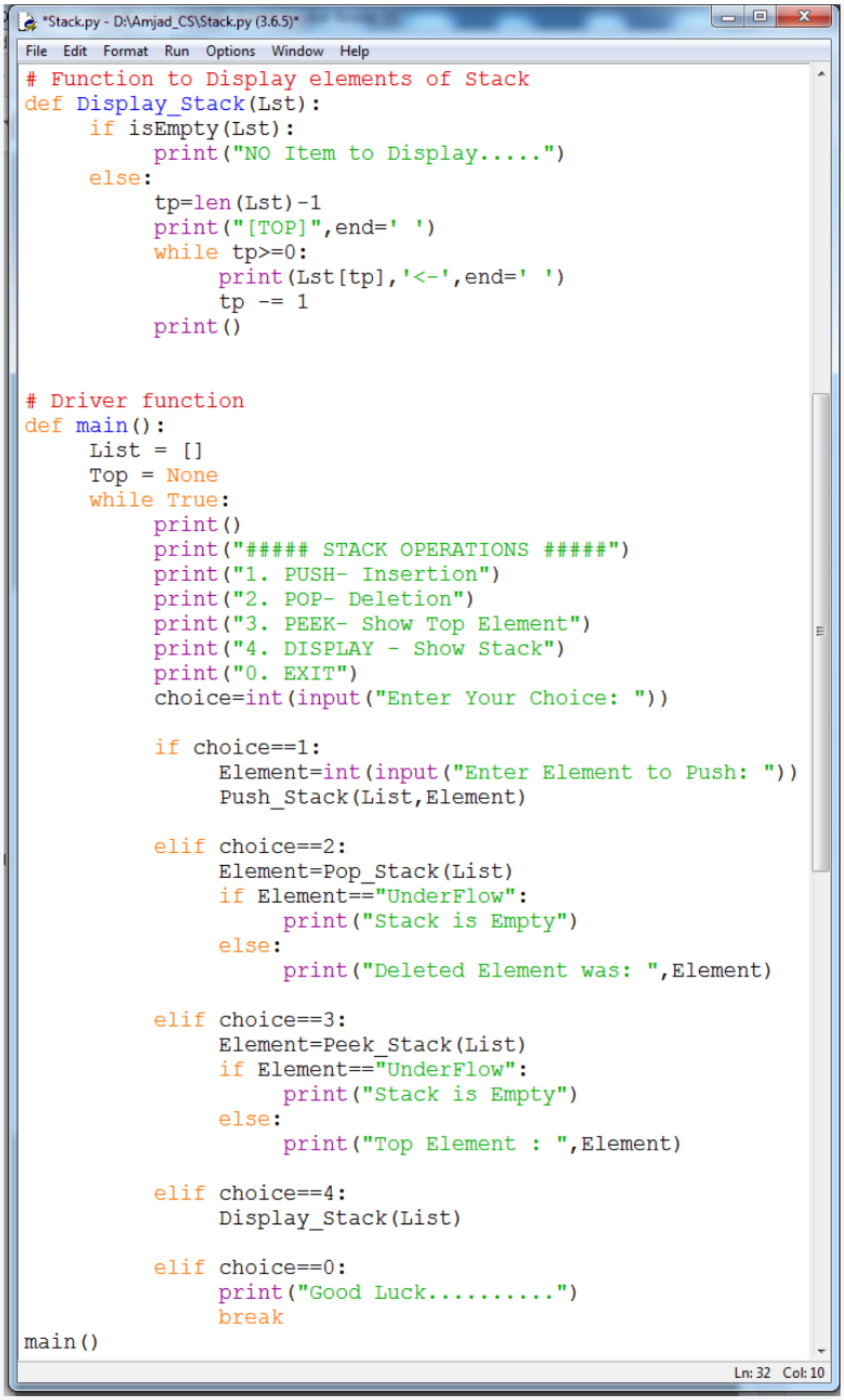
Chapter-10 Data Structure - II

Example: Implementing Stack In Python


Stack.py - D:\Amjad_CS\Stack.py (3.6.5)
File Edit Format Run Options Window Help
Ln: 37 Col: 23

```
# Created By: learnpython4cbse.com
# Stack using List

# Function to check Stack is empty or not
def isEmpty(Lst):
     if len(Lst)==0:
          return 1
     else:
          return 0

# Function to add (PUSH) elements in Stack
def Push_Stack(Lst,val):
     Lst.append(val)
     top=len(Lst)-1

# Function to Delete (POP) elements in Stack
def Pop_Stack(Lst):
     if isEmpty(Lst):
          return "UnderFlow"
     else:
          ele=Lst.pop()
          if len(Lst)==0:
               top=None
          else:
               top=len(Lst)-1
               return ele

# Function to Display Top element of Stack
def Peek_Stack(Lst):
     if isEmpty(Lst):
          return "UnderFlow"
     else:
               top=len(Lst)-1
               return Lst[top]
```

Chapter-10 Data Structure - II


Stack.py - D:\Amjad_CS\Stack.py (3.6.5)

```python
# Function to Display elements of Stack
def Display_Stack(Lst):
	if isEmpty(Lst):
		print("NO Item to Display.....")
	else:
		tp=len(Lst)-1
		print("[TOP]",end=' ')
		while tp>=0:
			print(Lst[tp],'<-',end=' ')
			tp -= 1
		print()


# Driver function
def main():
	List = []
	Top = None
	while True:
		print()
		print("##### STACK OPERATIONS #####")
		print("1. PUSH- Insertion")
		print("2. POP- Deletion")
		print("3. PEEK- Show Top Element")
		print("4. DISPLAY - Show Stack")
		print("0. EXIT")
		choice=int(input("Enter Your Choice: "))

		if choice==1:
			Element=int(input("Enter Element to Push: "))
			Push_Stack(List,Element)

		elif choice==2:
			Element=Pop_Stack(List)
			if Element=="UnderFlow":
				print("Stack is Empty")
			else:
				print("Deleted Element was: ",Element)

		elif choice==3:
			Element=Peek_Stack(List)
			if Element=="UnderFlow":
				print("Stack is Empty")
			else:
				print("Top Element : ",Element)

		elif choice==4:
			Display_Stack(List)

		elif choice==0:
			print("Good Luck..........")
			break
main()
```

Ln: 32 Col: 10

Chapter-10 Data Structure - II

OUTPUT

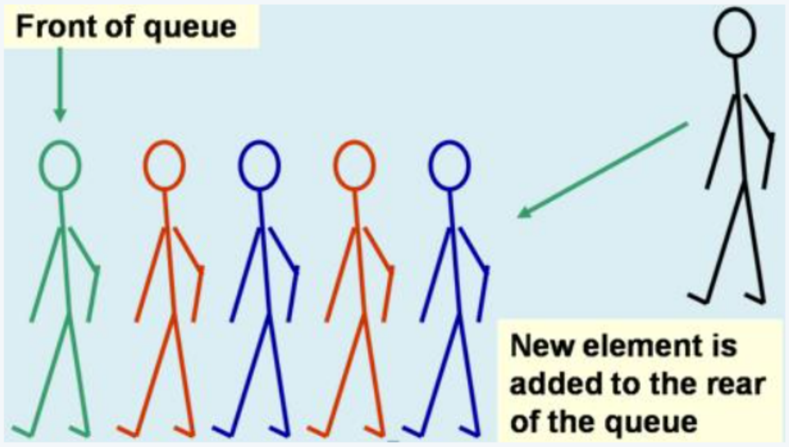
```
##### STACK OPERATIONS #####
1. PUSH- Insertion
2. POP- Deletion
3. PEEK- Show Top Element
4. DISPLAY - Show Stack
0. EXIT
Enter Your Choice: 1
Enter Element to Push: 20

##### STACK OPERATIONS #####
1. PUSH- Insertion
2. POP- Deletion
3. PEEK- Show Top Element
4. DISPLAY - Show Stack
0. EXIT
Enter Your Choice: 1
Enter Element to Push: 30

##### STACK OPERATIONS #####
1. PUSH- Insertion
2. POP- Deletion
3. PEEK- Show Top Element
4. DISPLAY - Show Stack
0. EXIT
Enter Your Choice: 1
Enter Element to Push: 88

##### STACK OPERATIONS #####
1. PUSH- Insertion
2. POP- Deletion
3. PEEK- Show Top Element
4. DISPLAY - Show Stack
0. EXIT
Enter Your Choice: 4
[TOP] 88 <- 30 <- 20 <-
```

```
##### STACK OPERATIONS #####
1. PUSH- Insertion
2. POP- Deletion
3. PEEK- Show Top Element
4. DISPLAY - Show Stack
0. EXIT
Enter Your Choice: 3
Top Element :  88

##### STACK OPERATIONS #####
1. PUSH- Insertion
2. POP- Deletion
3. PEEK- Show Top Element
4. DISPLAY - Show Stack
0. EXIT
Enter Your Choice: 2
Deleted Element was:  88

##### STACK OPERATIONS #####

1. PUSH- Insertion
2. POP- Deletion
3. PEEK- Show Top Element
4. DISPLAY - Show Stack
0. EXIT
Enter Your Choice: 4
[TOP] 30 <- 20 <-

##### STACK OPERATIONS #####
1. PUSH- Insertion
2. POP- Deletion
3. PEEK- Show Top Element
4. DISPLAY - Show Stack
0. EXIT
Enter Your Choice: 0
Good Luck..........
```

Chapter-10 Data Structure - II

2) Queue:

Queues are data structures that follow the **First In First Out (FIFO)** i.e. the first element that is added to the queue is the first one to be removed.



Real life examples

➔ Waiting in line

➔ Waiting on hold for tech support

➔ Applications related to Computer Science

➔ Round robin scheduling

➔ Key board buffer

QUEUE OPERATIONS:

➔ **Peek :** getting first value of QUEUE i.e. of FRONT position.

Queue[Front] ***# Front is an int storing index of first element of queue***

➔ **Enqueue:** addition of new item in QUEUE at REAR position.

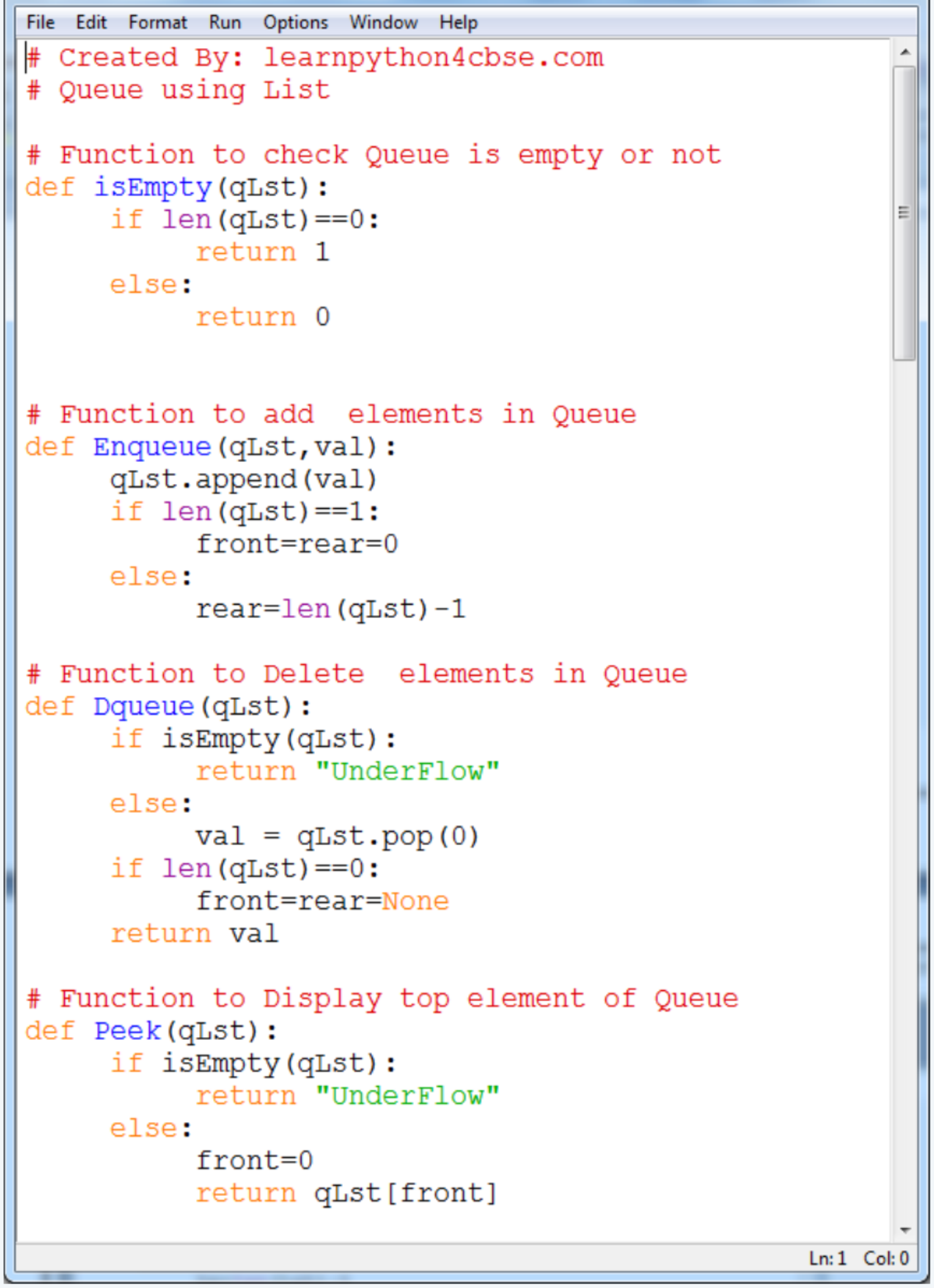
e.g. **Queue. append(Item)**

➔ **Dequeue:** removal of item from the beginning of QUEUE.

e.g. **Queue.pop(0)**

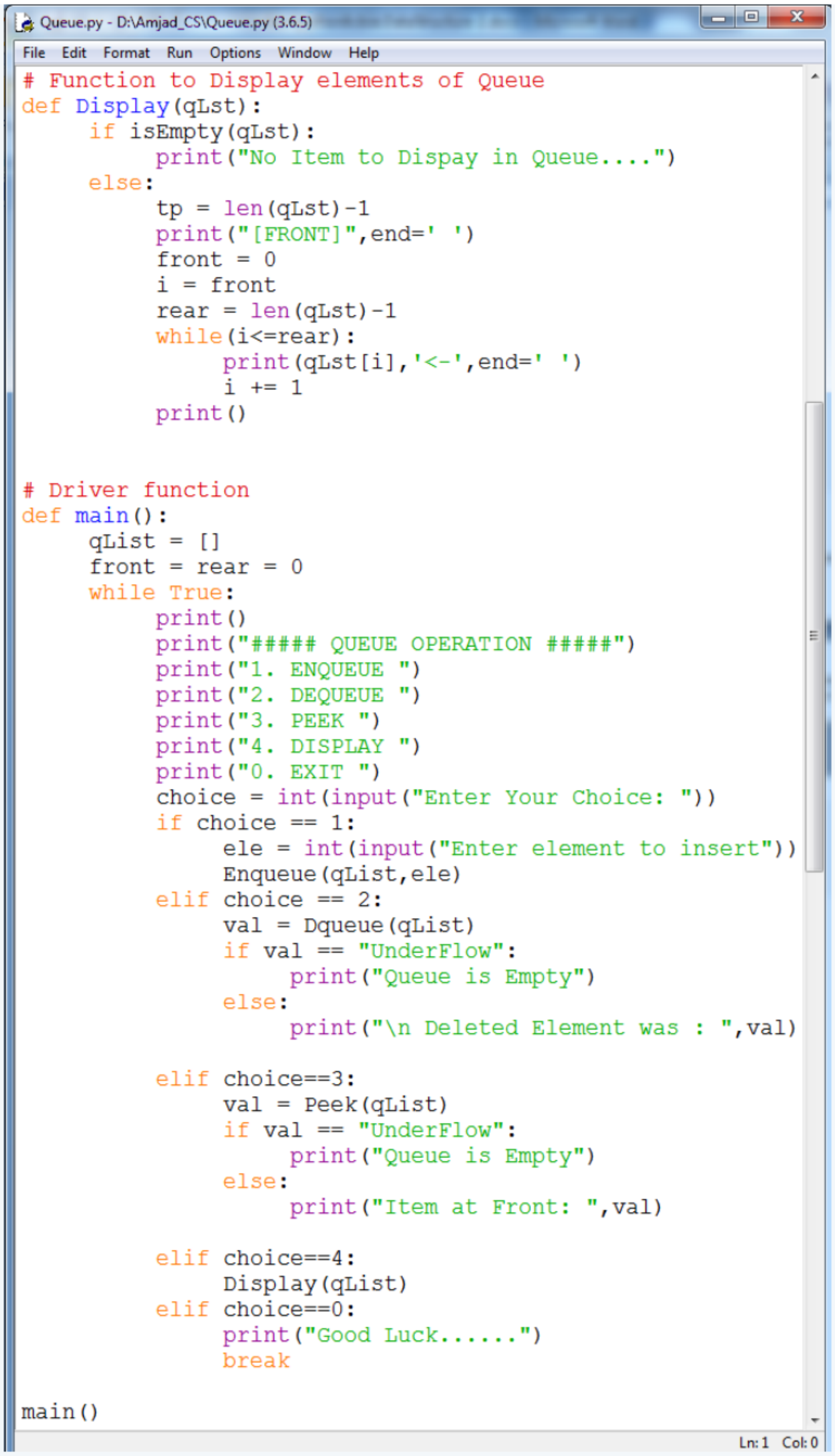
Chapter-10 Data Structure - II

Example: Implementation of Queue in Python



```
# Created By: learnpython4cbse.com
# Queue using List

# Function to check Queue is empty or not
def isEmpty(qLst):
    if len(qLst)==0:
        return 1
    else:
        return 0


# Function to add  elements in Queue
def Enqueue(qLst,val):
    qLst.append(val)
    if len(qLst)==1:
        front=rear=0
    else:
        rear=len(qLst)-1

# Function to Delete  elements in Queue
def Dqueue(qLst):
    if isEmpty(qLst):
        return "UnderFlow"
    else:
        val = qLst.pop(0)
    if len(qLst)==0:
        front=rear=None
    return val

# Function to Display top element of Queue
def Peek(qLst):
    if isEmpty(qLst):
        return "UnderFlow"
    else:
        front=0
        return qLst[front]
```

Chapter-10 Data Structure - II


Queue.py - D:\Amjad_CS\Queue.py (3.6.5)
File Edit Format Run Options Window Help
Ln: 1 Col: 0

```
# Function to Display elements of Queue
def Display(qLst):
    if isEmpty(qLst):
        print("No Item to Dispay in Queue....")
    else:
        tp = len(qLst)-1
        print("[FRONT]",end=' ')
        front = 0
        i = front
        rear = len(qLst)-1
        while(i<=rear):
            print(qLst[i],'<-',end=' ')
            i += 1
        print()


# Driver function
def main():
    qList = []
    front = rear = 0
    while True:
        print()
        print("##### QUEUE OPERATION #####")
        print("1. ENQUEUE ")
        print("2. DEQUEUE ")
        print("3. PEEK ")
        print("4. DISPLAY ")
        print("0. EXIT ")
        choice = int(input("Enter Your Choice: "))
        if choice == 1:
            ele = int(input("Enter element to insert"))
            Enqueue(qList,ele)
        elif choice == 2:
            val = Dqueue(qList)
            if val == "UnderFlow":
                print("Queue is Empty")
            else:
                print("\n Deleted Element was : ",val)

        elif choice==3:
            val = Peek(qList)
            if val == "UnderFlow":
                print("Queue is Empty")
            else:
                print("Item at Front: ",val)

        elif choice==4:
            Display(qList)
        elif choice==0:
            print("Good Luck......")
            break

main()
```

Chapter-10 Data Structure - II

OUTPUT:

```
##### QUEUE OPERATION #####
1. ENQUEUE
2. DEQUEUE
3. PEEK
4. DISPLAY
0. EXIT
Enter Your Choice: 1
Enter element to insert20

##### QUEUE OPERATION #####
1. ENQUEUE
2. DEQUEUE
3. PEEK
4. DISPLAY
0. EXIT
Enter Your Choice: 1
Enter element to insert25

##### QUEUE OPERATION #####
1. ENQUEUE
2. DEQUEUE
3. PEEK
4. DISPLAY
0. EXIT
Enter Your Choice: 1
Enter element to insert90

##### QUEUE OPERATION #####
1. ENQUEUE
2. DEQUEUE
3. PEEK
4. DISPLAY
0. EXIT
Enter Your Choice: 4
[FRONT] 20 <- 25 <- 90 <-
```

```
##### QUEUE OPERATION #####
1. ENQUEUE
2. DEQUEUE
3. PEEK
4. DISPLAY
0. EXIT
Enter Your Choice: 3
Item at Front:  20

##### QUEUE OPERATION #####
1. ENQUEUE
2. DEQUEUE
3. PEEK
4. DISPLAY
0. EXIT
Enter Your Choice: 2

 Deleted Element was :  20

##### QUEUE OPERATION #####
1. ENQUEUE
2. DEQUEUE
3. PEEK
4. DISPLAY
0. EXIT
Enter Your Choice: 4
[FRONT] 25 <- 90 <-

##### QUEUE OPERATION #####
1. ENQUEUE
2. DEQUEUE
3. PEEK
4. DISPLAY
0. EXIT
Enter Your Choice: 0
Good Luck......
```