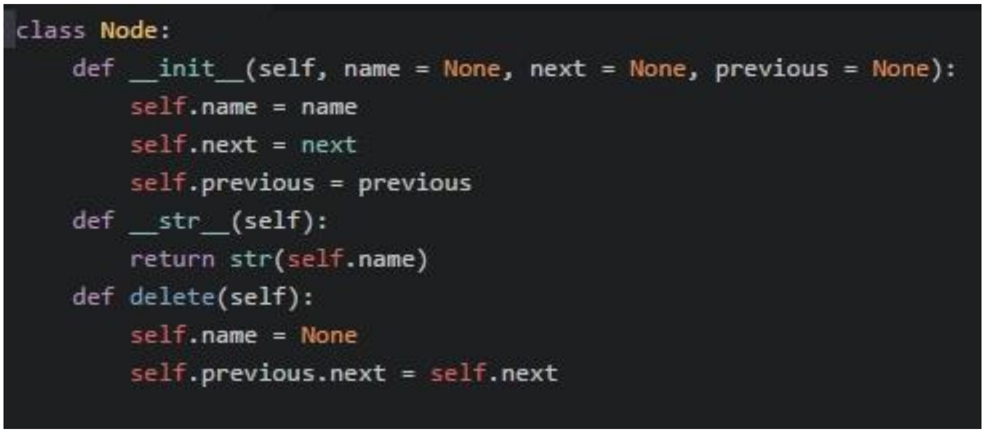
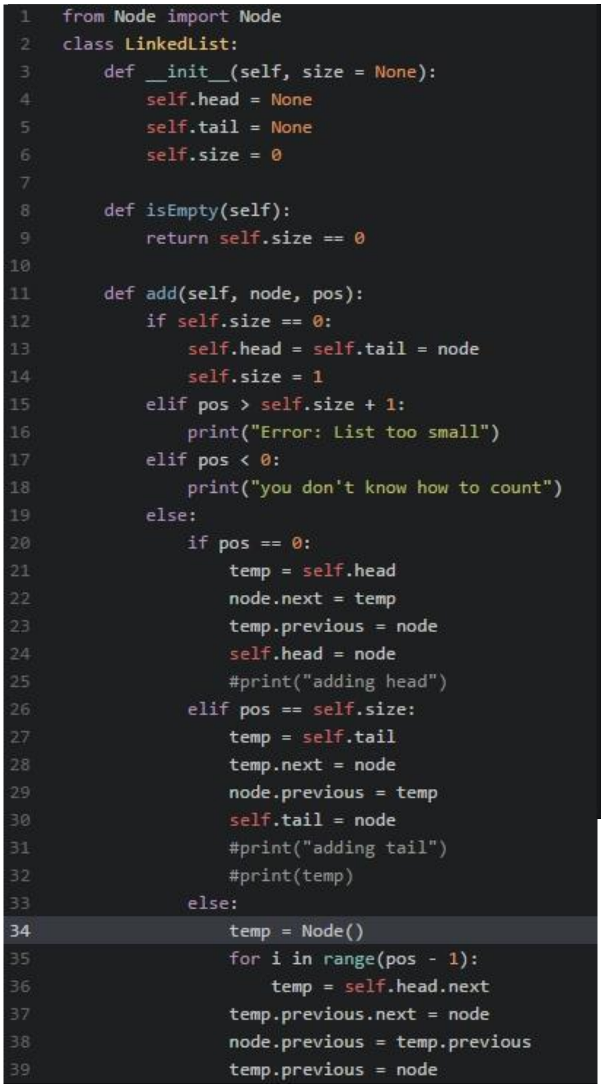
Appendix

NODE.PY

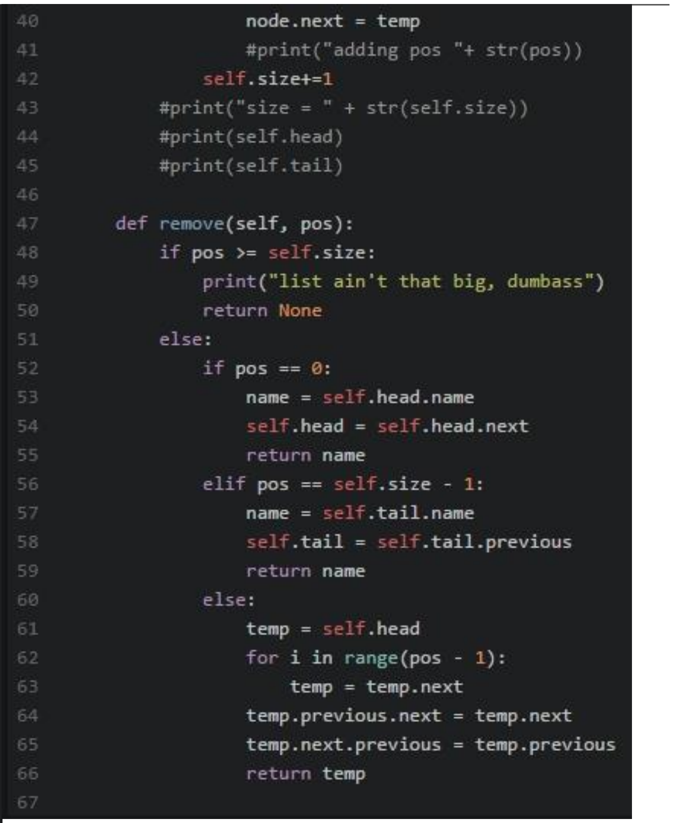


```
class Node:
    def __init__(self, name = None, next = None, previous = None):
        self.name = name
        self.next = next
        self.previous = previous
    def __str__(self):
        return str(self.name)
    def delete(self):
        self.name = None
        self.previous.next = self.next
```

LINKEDLIST.PY

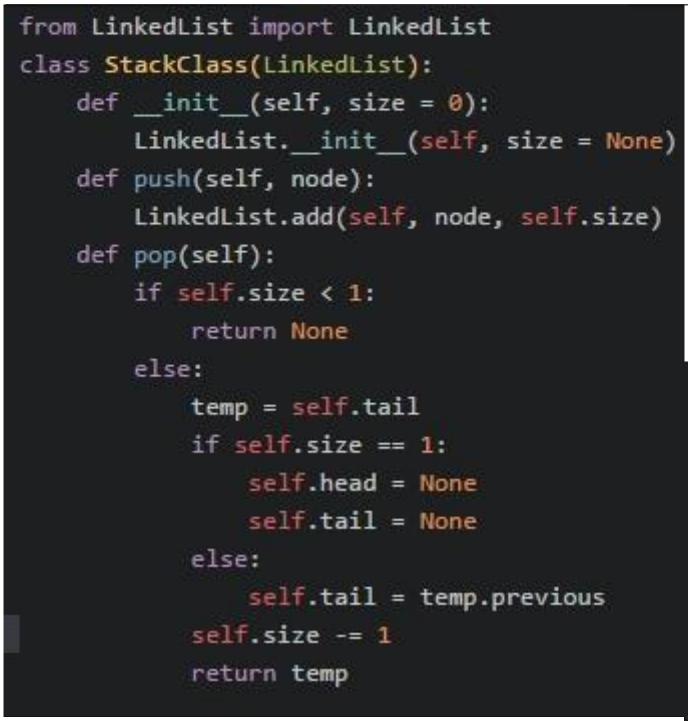


```
from Node import Node
class LinkedList:
    def __init__(self, size = None):
        self.head = None
        self.tail = None
        self.size = 0

    def isEmpty(self):
        return self.size == 0

    def add(self, node, pos):
        if self.size == 0:
            self.head = self.tail = node
            self.size = 1
        elif pos > self.size + 1:
            print("Error: List too small")
        elif pos < 0:
            print("you don't know how to count")
        else:
            if pos == 0:
                temp = self.head
                node.next = temp
                temp.previous = node
                self.head = node
                #print("adding head")
            elif pos == self.size:
                temp = self.tail
                temp.next = node
                node.previous = temp
                self.tail = node
                #print("adding tail")
                #print(temp)
            else:
                temp = Node()
                for i in range(pos - 1):
                    temp = self.head.next
                temp.previous.next = node
                node.previous = temp.previous
                temp.previous = node
```



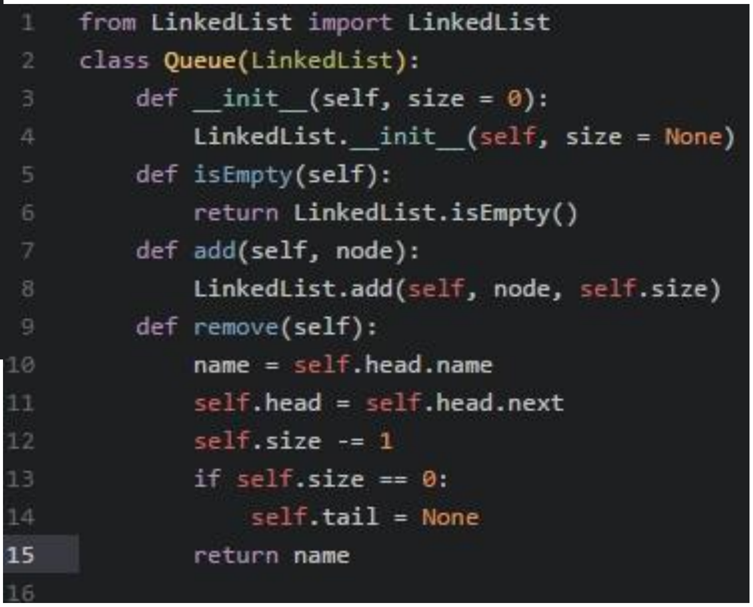
```
                node.next = temp
                #print("adding pos "+ str(pos))
            self.size+=1
        #print("size = " + str(self.size))
        #print(self.head)
        #print(self.tail)

    def remove(self, pos):
        if pos >= self.size:
            print("list ain't that big, dumbass")
            return None
        else:
            if pos == 0:
                name = self.head.name
                self.head = self.head.next
                return name
            elif pos == self.size - 1:
                name = self.tail.name
                self.tail = self.tail.previous
                return name
            else:
                temp = self.head
                for i in range(pos - 1):
                    temp = temp.next
                temp.previous.next = temp.next
                temp.next.previous = temp.previous
                return temp
```

StackClass.py



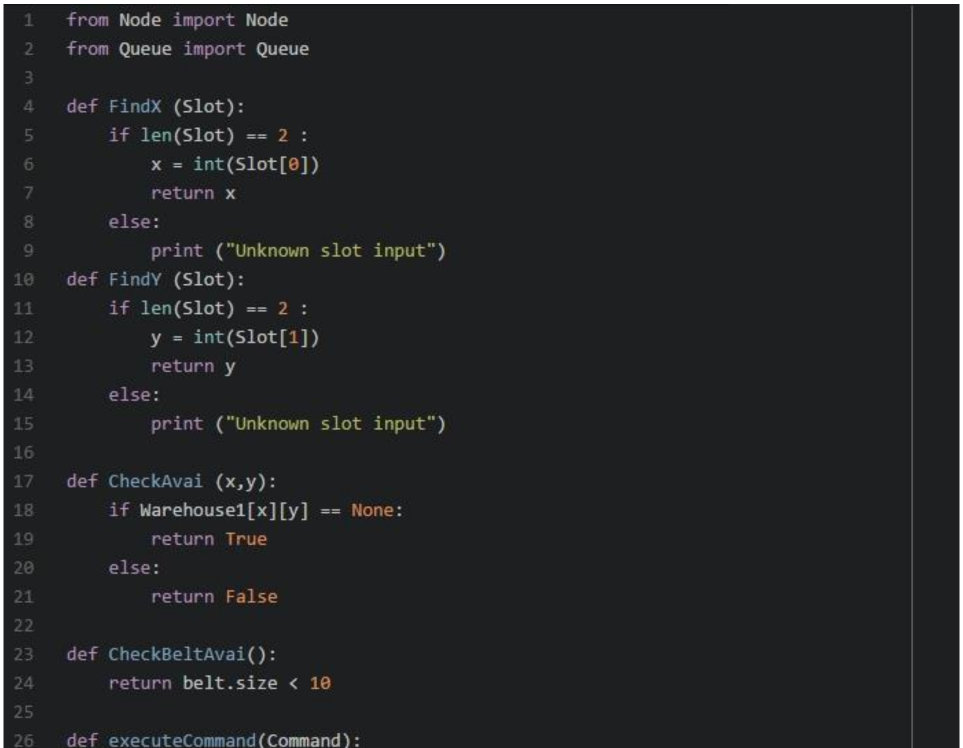
```python
from LinkedList import LinkedList
class StackClass(LinkedList):
    def __init__(self, size = 0):
        LinkedList.__init__(self, size = None)
    def push(self, node):
        LinkedList.add(self, node, self.size)
    def pop(self):
        if self.size < 1:
            return None
        else:
            temp = self.tail
            if self.size == 1:
                self.head = None
                self.tail = None
            else:
                self.tail = temp.previous
            self.size -= 1
            return temp
```

Queue.py

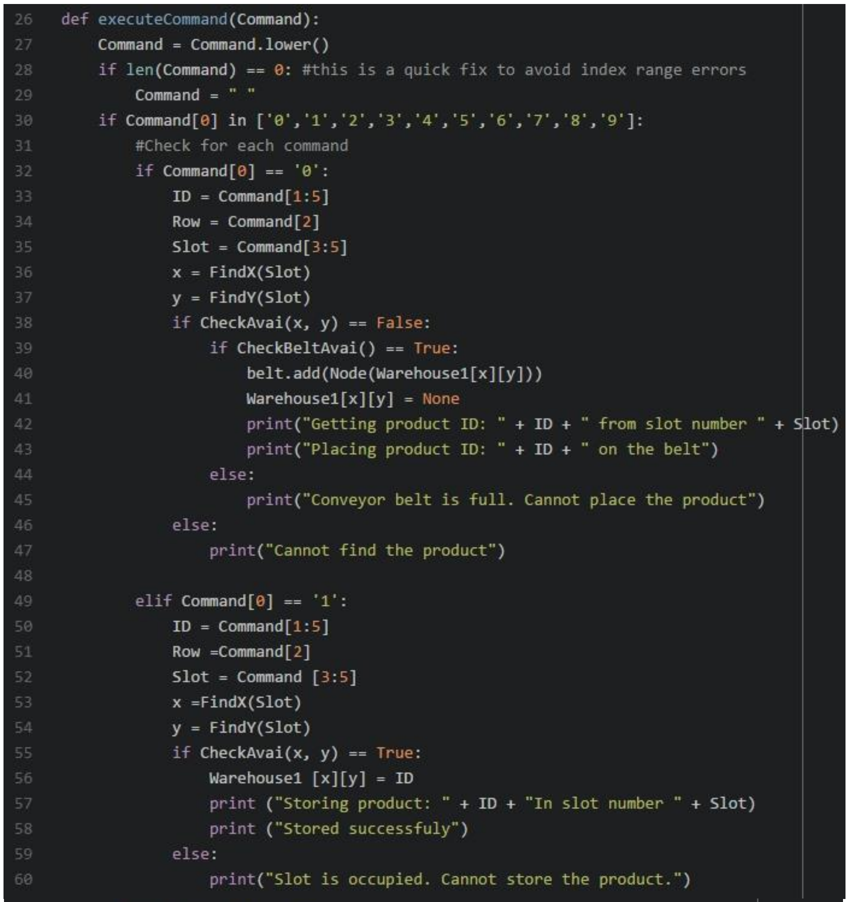


```python
from LinkedList import LinkedList
class Queue(LinkedList):
    def __init__(self, size = 0):
        LinkedList.__init__(self, size = None)
    def isEmpty(self):
        return LinkedList.isEmpty()
    def add(self, node):
        LinkedList.add(self, node, self.size)
    def remove(self):
        name = self.head.name
        self.head = self.head.next
        self.size -= 1
        if self.size == 0:
            self.tail = None
        return name

```

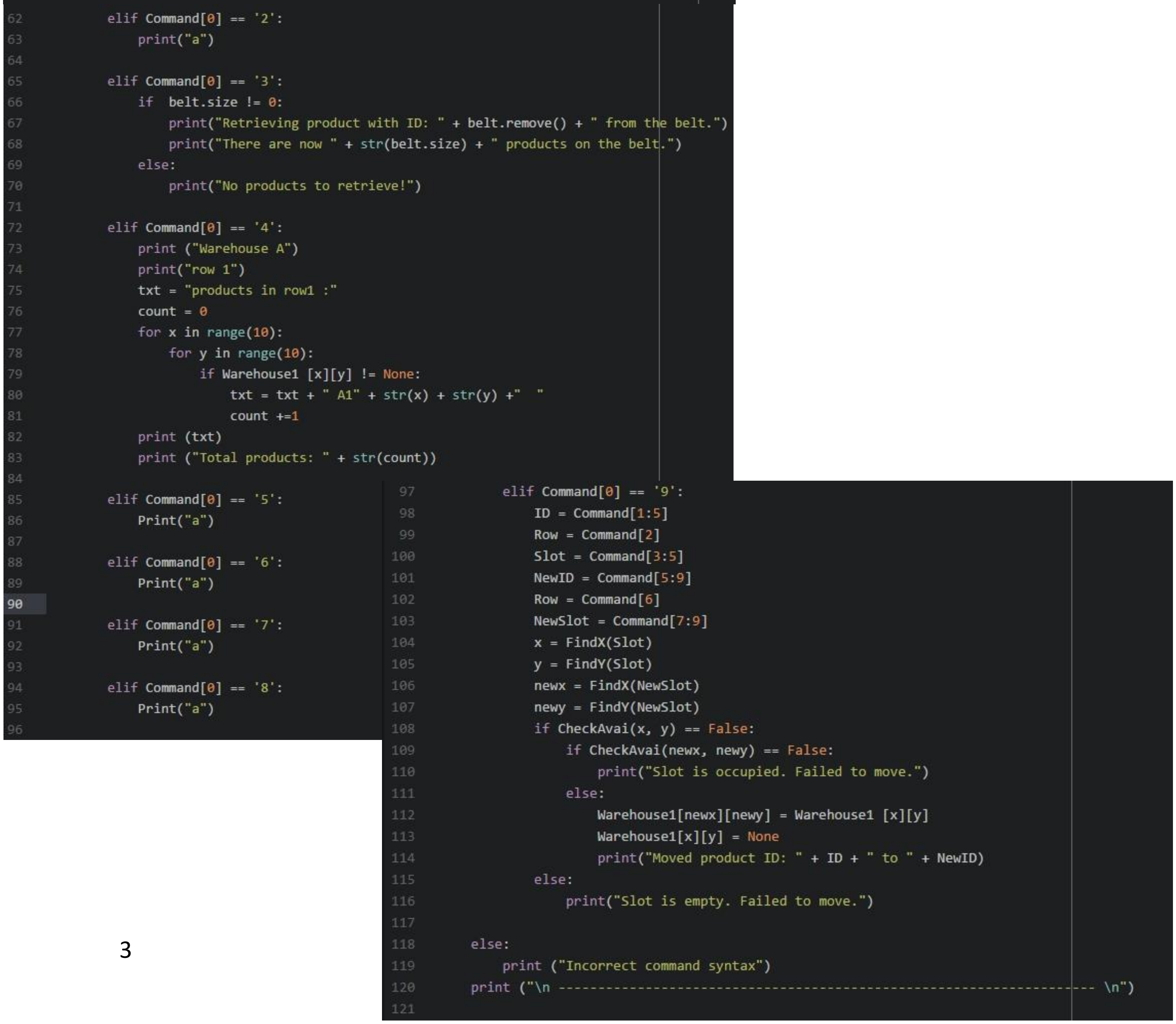
Warehouse.py



```python
from Node import Node
from Queue import Queue

def FindX (Slot):
    if len(Slot) == 2 :
        x = int(Slot[0])
        return x
    else:
        print ("Unknown slot input")
def FindY (Slot):
    if len(Slot) == 2 :
        y = int(Slot[1])
        return y
    else:
        print ("Unknown slot input")

def CheckAvai (x,y):
    if Warehouse1[x][y] == None:
        return True
    else:
        return False

def CheckBeltAvai():
    return belt.size < 10

def executeCommand(Command):
```

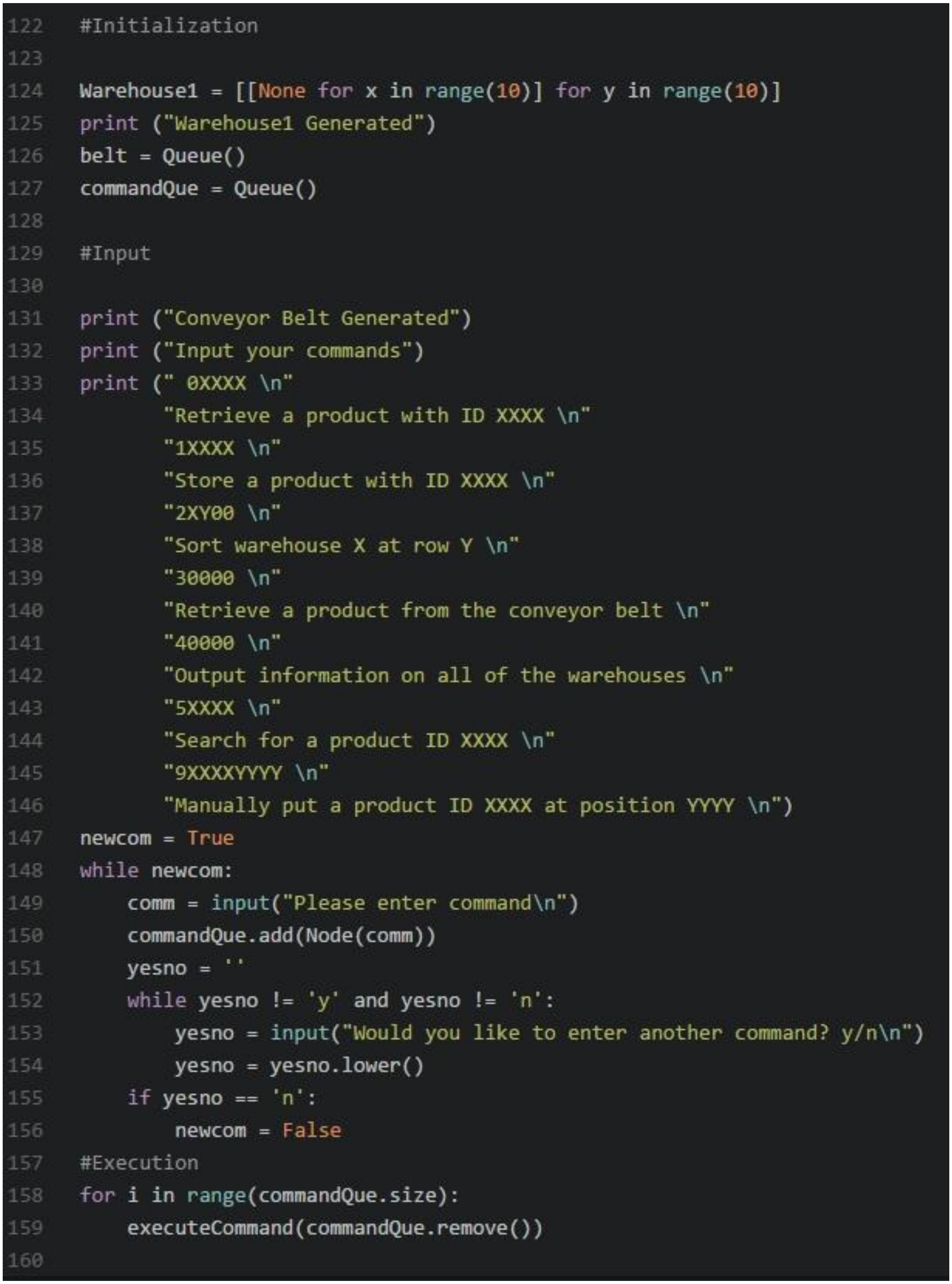


```python
def executeCommand(Command):
    Command = Command.lower()
    if len(Command) == 0: #this is a quick fix to avoid index range errors
        Command = " "
    if Command[0] in ['0','1','2','3','4','5','6','7','8','9']:
        #Check for each command
        if Command[0] == '0':
            ID = Command[1:5]
            Row = Command[2]
            Slot = Command[3:5]
            x = FindX(Slot)
            y = FindY(Slot)
            if CheckAvai(x, y) == False:
                if CheckBeltAvai() == True:
                    belt.add(Node(Warehouse1[x][y]))
                    Warehouse1[x][y] = None
                    print("Getting product ID: " + ID + " from slot number " + Slot)
                    print("Placing product ID: " + ID + " on the belt")
                else:
                    print("Conveyor belt is full. Cannot place the product")
            else:
                print("Cannot find the product")

        elif Command[0] == '1':
            ID = Command[1:5]
            Row =Command[2]
            Slot = Command [3:5]
            x =FindX(Slot)
            y = FindY(Slot)
            if CheckAvai(x, y) == True:
                Warehouse1 [x][y] = ID
                print ("Storing product: " + ID + "In slot number " + Slot)
                print ("Stored successfuly")
            else:
                print("Slot is occupied. Cannot store the product.")
```



```python
        elif Command[0] == '2':
            print("a")

        elif Command[0] == '3':
            if  belt.size != 0:
                print("Retrieving product with ID: " + belt.remove() + " from the belt.")
                print("There are now " + str(belt.size) + " products on the belt.")
            else:
                print("No products to retrieve!")

        elif Command[0] == '4':
            print ("Warehouse A")
            print("row 1")
            txt = "products in row1 :"
            count = 0
            for x in range(10):
                for y in range(10):
                    if Warehouse1 [x][y] != None:
                        txt = txt + " A1" + str(x) + str(y) +"  "
                        count +=1
            print (txt)
            print ("Total products: " + str(count))

        elif Command[0] == '5':
            Print("a")

        elif Command[0] == '6':
            Print("a")

        elif Command[0] == '7':
            Print("a")

        elif Command[0] == '8':
            Print("a")
```

```python
        elif Command[0] == '9':
            ID = Command[1:5]
            Row = Command[2]
            Slot = Command[3:5]
            NewID = Command[5:9]
            Row = Command[6]
            NewSlot = Command[7:9]
            x = FindX(Slot)
            y = FindY(Slot)
            newx = FindX(NewSlot)
            newy = FindY(NewSlot)
            if CheckAvai(x, y) == False:
                if CheckAvai(newx, newy) == False:
                    print("Slot is occupied. Failed to move.")
                else:
                    Warehouse1[newx][newy] = Warehouse1 [x][y]
                    Warehouse1[x][y] = None
                    print("Moved product ID: " + ID + " to " + NewID)
            else:
                print("Slot is empty. Failed to move.")

    else:
        print ("Incorrect command syntax")
    print ("\n ------------------------------------------------------------------ \n")
```



```
#Initialization

Warehouse1 = [[None for x in range(10)] for y in range(10)]
print ("Warehouse1 Generated")
belt = Queue()
commandQue = Queue()

#Input

print ("Conveyor Belt Generated")
print ("Input your commands")
print (" 0XXXX \n"
       "Retrieve a product with ID XXXX \n"
       "1XXXX \n"
       "Store a product with ID XXXX \n"
       "2XY00 \n"
       "Sort warehouse X at row Y \n"
       "30000 \n"
       "Retrieve a product from the conveyor belt \n"
       "40000 \n"
       "Output information on all of the warehouses \n"
       "5XXXX \n"
       "Search for a product ID XXXX \n"
       "9XXXXYYYY \n"
       "Manually put a product ID XXXX at position YYYY \n")
newcom = True
while newcom:
    comm = input("Please enter command\n")
    commandQue.add(Node(comm))
    yesno = ''
    while yesno != 'y' and yesno != 'n':
        yesno = input("Would you like to enter another command? y/n\n")
        yesno = yesno.lower()
    if yesno == 'n':
        newcom = False
#Execution
for i in range(commandQue.size):
    executeCommand(commandQue.remove())

```