

Infosys  
Springboard

Explore

EN

What do you want to learn?

**Data Structures and Algorithms using Python - Part 1**

In this course, we introduce the basic concepts of Data Structures in programming using 'Python' language constructs. In common words, Data Structures are storage CO...More

Start

**Learning Progress**

Status: Completed

**Completion Certificate**

Click on Generate Certificate button to claim your completion certificate

Generate Certificate

**Overview** Contents Discussions

**What you will learn**

Introduce fundamentals of Data Structures. Discuss frequently used linear data structures like Stack, Queue, List etc. Discuss the commonly performed operations on linear data structures like insert, update, delete etc. Provide insights on non - linear data structures.

**Skills you will gain**

**Authors/Creators**

MP Madhura Prasad

AA Avinash Ajay Malawade

SG Saisree G Ramanaiah

RN Rishabh Nanda

MM Malathi M

Show Curators/Contacts

**At a glance**

Course

29h 27m

Beginner Level

Free

Infosys Wingspan

EN

datastructures algorithms dea

Thank you. Your test submitted.

You have cleared this assessment.

Obtained Percentage	Obtained Marks
100 %	13 / 13

Best Attempt Score:100 % on 23-03-2025

Given the following stack:

(Top→Bottom)

7→12→1→5→4→2

What will be the updated content of the above stack when passed to the below Python function?

Assumption: All the references to the necessary files are available

```
def fun(stack1):  
    if(stack1.is_empty()):  
        return  
    else:  
        s1=Stack(7)  
        while(not stack1.is_empty()):  
            s1.push(stack1.pop())  
            s1.push(5)  
            s1.push(s1.pop()+s1.pop())  
        while(not s1.is_empty()):  
            stack1.push(s1.pop())
```

Note: The options show the stack content from top to bottom

- ☐ 12→17→6→10→9→17
- ☐ 12→17→16→10→9→7
- ☐ 12→17→6→10→19→7
- ☒ 12→17→6→10→9→7

Warning

This operation is disabled.

Ok

Consider the below Python code:

```
def process(input_stack):  
    output_stack=Stack(5)  
    count=0  
    while(not input_stack.is_empty()):  
        temp=input_stack.pop()  
        for i in temp:  
            count+=1  
        output_stack.push(str(count)+temp)  
    return output_stack
```

```
input_stack=Stack(5)  
input_stack.push("India")  
input_stack.push("Australia")  
input_stack.push("England")  
input_stack.push("SouthAfrica")
```

```
process(input_stack).display()
```

What will be the status of output\_stack after execution of a

Assumption: All the references to the necessary files are available

Note: Consider that the elements of stack in the options are shown

- ☐ 11SouthAfrica-->18England-->27Australia-->32India
- ☐ 5India-->9Australia-->7England-->11SouthAfrica
- ☐ 11SouthAfrica-->7England-->9Australia-->5India
- ☒ 32India-->27Australia-->18England-->11SouthAfrica

Wal

This

Consider the below Python code:

Assumption: All the references to the necessary files are available

```
def func(input_queue):  
    q1=Queue(input_queue.max_size)  
    q2=Queue(input_queue.max_size)  
    for i in range(input_queue.front,input_queue.rear+1):  
        item=input_queue.dequeue()  
        if(item>10):  
            q1.enqueue(item)  
        else:  
            q2.enqueue(item)
```

```
input_queue=Queue(5)
```

```
input_queue.enqueue(12)  
input_queue.enqueue(10)  
input_queue.enqueue(15)  
input_queue.enqueue(1)  
input_queue.enqueue(20)
```

```
func(input_queue)
```

What is the status of Queues q1 and q2 after execution of above code?

The order of elements in options are from Front to Rear

- ☐ q1 contains: 12-->20-->15 and q2 contains: 10-->1
- ☐ q1 contains: 12-->15-->20 and q2 contains: 1-->10
- ☒ q1 contains: 12-->15-->20 and q2 contains: 10-->1
- ☐ q1 contains: 12-->10-->15-->20 and q2 contains: 1

Consider a treasure hunt game in which you are handed a clue that leads you to another clue and so on until you find the treasure. The first clue links (or points) to the location of next clue. The second clue links to the location of third, and so on.

Which data structure can be optimally used to model the treasure hunt game?

- ☐ Stack
- ☒ Linked List
- ☐ Graph
- ☐ Queue

Warning

This operation is disabled.

Ok



Given the following linked list:

1->4->6->7->9

What would be the state of the linked list when the following Python function is invoked by passing the head node of the above linked list and 5?

```
def fun(head,n):
    temp=head
    if(head==None):
        return
    else:
        i=1
        while(i<n):
            val=temp.get_data()
            temp=temp.get_next()
            head.set_data(val)
            i+=1
        temp1=head.get_data()
        temp.set_data(temp1)
```

Warning

This operation is disabled.

Ok

Assumption: All the references to the necessary files are available. Method get\_next() returns the link to the next node. Method get\_data() returns data of the node.

- ☐ 1->4->6->7->7
- ☒ 7->4->6->7->7
- ☐ 1->4->6->7->9
- ☐ 7->4->6->7->9

Given the following Linked List: 12->121->112->111->222

What would be the state of the linked list when the following Python function is invoked by passing the head node of the above linked list?

Assumption: All the references to the necessary files are present.

```
def fun(head):  
    temp=head  
    if(head==None):  
        return  
    else:  
        while(temp.next!=None):  
            temp.data+=temp.next.data  
            temp=temp.next
```

Warning

This operation is disabled.

Ok

- ☐ 133->233->223->111->222
- ☐ 133->233->112->333->222
- ☒ 133->233->223->333->222
- ☐ 133->233->223->333->355



Consider a customer Service hotline. Sometimes when a call comes in and there are not enough operators to handle the call, it is placed on hold. When the next call comes in, it is also placed on hold and this continues till the operator becomes available.

When the operator becomes available, the first call that was put on hold gets assigned to him/her.

Which of the below data structure is optimal to represent this "hotline"?

- ☐ Tree
- ☐ Graph
- ☒ Queue
- ☐ Stack

### Warning

This operation is disabled.

Ok

What will be the output of the below Python function when it is invoked by passing n as 4?

Assumption: All the references to the necessary files are available and n is a non-zero value.

```
def fun(n):  
    q=Queue(n)  
    if(n==0):  
        return 1  
    else:  
        for i in range(1,n+1):  
            q.enqueue(i*i)  
        temp=1  
        while(not q.is_empty()):  
            temp+=q.dequeue()  
        return temp
```

Warning

This operation is disabled.

Ok

- ☒ 31
- ☐ 30
- ☐ 25
- ☐ Function fun() would always return 1

What does the below Python function do using Queue data structure?

Assumption: All the references to the necessary files are available and n is a non-zero value.

```
def fun(n):  
    q=Queue(n) # This statement creates a queue of size n  
    if(n==0):  
        return 1  
    else:  
        for i in range(1,n+1):  
            q.enqueue(i)  
        temp=1  
        while(not q.is_empty()):  
            temp*=q.dequeue()  
        return temp
```

Warning

This operation is disabled.

Ok

- ☐ Function fun() returns the binary equivalent of a number 'n'
- ☒ Function fun() returns the factorial of a number 'n'
- ☐ Function fun() returns the sum of first 'n' numbers
- ☐ Function fun() would always return 1

Which of the following statements are FALSE regarding Array data structure?

Select TWO correct options.

- ☐ Deleting an element from the beginning of an array
- ☐ Inserting an element at the end of an array
- ☒ Elements of the array cannot be accessed randomly
- ☒ Elements of an array cannot be accessed sequentially

## Warning

This operation is disabled

Which of the following statements are TRUE with respect to the Array Data Structure?  
Select THREE CORRECT options.

- ☐ Inserting an element at the end requires shifting of all elements
- ☒ Elements of an array can be accessed randomly
- ☒ Inserting an element at beginning requires shifting of all elements
- ☒ Deleting an element from beginning requires shifting of all elements

Warning

This operation is disabled.

Given the following linked list:

12->34->26->67->98

What would be the state of the linked list when the following Python function is invoked by passing the head node of the above linked list and 3?

Assumption: All the references to the necessary files are available

```
def fun(head, n):  
    temp=head  
    if(head==None):  
        return  
    else:  
        i=1  
        while(i!=n):  
            temp=temp.next  
            i+=1  
        temp1=head.data  
        head.data=temp.data  
        temp.data=temp1
```

Warning

This operation is disabled.

Ok

- ☐ 12->34->26->67->98
- ☒ 26->34->12->67->98
- ☐ 67->34->26->12->98
- ☐ 98->34->26->67->12



Given the following stack:

(Top-->Bottom)

7->12->1->5->4->2

What will be the output of the below Python function when it is invoked by passing the above stack?

Assumption: All the references to the necessary files are available

```
def fun(stack1):  
    if(stack1.is_empty()):  
        return  
    else:  
        s1=Stack(4)  
        while(not stack1.is_empty()):  
            s1.push(stack1.pop())  
            s1.push(stack1.pop())  
            s1.push(s1.pop()*s1.pop())  
        return s1
```

Note: The options show the stack output from top

Warning

This operation is disabled.

Ok

- ☐ 84->8->5
- ☐ 5->84->8
- ☒ 8->5->84
- ☐ No change