Algorithms and Data Structures : Week 1

Representations of data structures

Multi-Dimensional Array Tasks

Sum Columns of Table

Write a function to sum the columns of a 2 dimensional array (i.e. a list of lists). The function should work for any size table (providing it is regular in shape). For example, with:

```
table1 = [  [1, 2, 3, 4],  [4, 5, 6, 7],  [8, 9,10,11] ]
we get:
>>> sumCol(table1,0)
13
>>> sumCol(table1,1)
16
```

Save the Princess

Maurizio needs to move through the grid to rescue Princess Pineapple but being a lazy programmer you don't want to control him manually). You will need to build a program that will analyse the state of the game world and output the right set of commands….

```
from random import randint
size=2
while size%2==0:
    size=int(input("enter the grid size - must be an odd number: "))
grid=[["-" for x in range(size)] for y in range(size)]
mid=(size//2)
grid[mid][mid]="m"
grid[randint(0,size)][randint(0,size)]="p"

for row in grid:
    for e in row:
        print (e, end='')
    print()

print("Maurizio needs to rescue the Princess")
#you need to write the code to solve the puzzle
# It should output the directions Maurizio needs to take to get to
Princess Pineapple
# Directions should be output in the form "UP" "DOWN" "LEFT" "RIGHT"
# e.g. "UP" "UP" "LEFT" "LEFT" "LEFT"
```

Stacks

The operations of a stack are:

- `push(a)` – put item 'a' on the top of the stack
- `a = pop()` – remove the top item from the stack

Consider the following sequence of operations:

```
1. push(8)
2. push(2)
3. a = pop() ; b = pop() ; push(b / a)
4. push(5)
5. a = pop() ; b = pop() ; push(b + a)
6. push(3)
7. push(1)
8. a = pop() ; b = pop() ; push(b - a)
9. a = pop() ; b = pop() ; push(b * a)
```

The following table shows the state of the stack after each step, completed up to step 4. Check these steps and complete the rest of the table.

		2		5					
	8	8	4	4					
Step:	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**

Stacks Exercises

Given the list
`stack=["bob","James","","","","","","","",""]`

to represent an array
and the variable
`topOfStack=2`

Write two procedures

`myPush` that takes a value and puts it in the appropriate place in stack

`myPop` that removes the last (top) value from the stack and returns it

all errors should be appropriately handled

`stack` and `topofstack` may be used as global variables in this assignment

easy, medium and hard code skeletons are available

Challenge 1 - Medium

Can you rewrite this not need the topOfStack variable

Challenge 2 - HARD

Can you rewrite this to not need any global variables

Challenge 3 –

Can you rewrite this to create a stack class and methods

Queues

Given the queue
`queue=["bob","James","","","","","","","",""]`
to represent an array
and the variable
`backofqueue=2`

Write two procedures

enqueue

```
    #put the item at the back of the queue
    # change back of queue value
```

dequeue

```
    #return first value
    #update rest of queue..
```

- all errors should be appropriately handled
- queue and backofqueue may be used as global variables in this assignment
- easy, medium and hard code skeletons are available

Challenge 1 - Medium

Can you rewrite this to not need the backofqueue variable

Challenge 2 - HARD

Can you rewrite this to not need any global variables

Challenge 3 –

Can you rewrite this to create a queue class and methods

Priority Queues

Given the queue
`queue=[("bob",1),("James",1),"","","","","","","","",""]`
where each item is stored as a tuple of (value, priority)

and the variable
`backofqueue=2`

Write the dequeue procedure

```
dequeue

    #find the highest priority item in the queue and return it,
    #update queue
```

all errors should be appropriately handled

queue and backofqueue may be used as global variables in this assignment

easy, medium and hard code skeletons are available

Circular Queues

Given the queue
`queue=["","","","","","","Dave","Eve","Fred",""]`
and the variables
`backOfQueue=9`
`frontOfQueue=6`

Write the `enqueue` procedure
#update the back of Queue and wrap round if needed

Write the `dequeue` procedure
#store the value at the front of the queue
#clear the space and update the pointer
#return the stored value

dequeue

    #find the highest priority item in the queue and return it, update queue

Full and empty queues should be detected

Queue, frontOfQueue and backofqueue may be used as global variables in this assignment

easy, medium and hard code skeletons are available

Challenge 1

Adapt your solution to create a circular priority queue

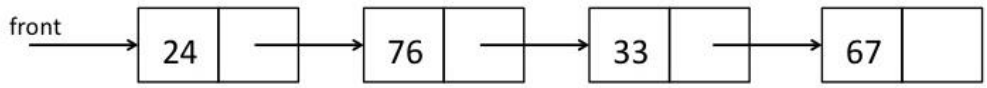
Challenge 2

Rewrite as a class with methods

Linked Lists

List represented by a linked list

The following picture shows a linked list. What is the (Python) list represented by the linked list?



Operations on a linked list

Redraw the picture of the linked list following the following cases:

1. Case 1: the number 55 is appended to the end of the list.
2. Case 2: the number 33 is removed from the list.

Step by step through the list

To append 55 at the end of the list, it is necessary to walk down the list to find the last cell. The steps are as follows:

1) Make the current pointer (arrow) 'front'.
2) Check if the current pointer is null (points at nothing).
   a) If not, the current pointer ⍰ the next pointer of the current cell
   b) Repeat
3) Create a new cell, containing 55
4) Make the next pointer of the current cell point to the new cell

Write a similar set of steps for removing the number 33.