1) what are the stacks, and what are the main
charectristies of stack.
⇒ STACK (abstract datatype that serves a collection of a data)
-> LIFO Rule
-> top is used to keep or remove the elements
→ main operations Push, pop L → user defined
adding Removing
Is empty } conditions to be checked. Is full adding strongs
peek (or) Top (to know the top/last element)
NORKING OF A STACK
Stack is empty then top value =-1.
Top=-1.
CHARECTERISTICS :-
-> Implemented through array/ linked list.
-> Towers of Hanoi, tree traversals
-> over flow - stack full
-> under flow - stack empty.
Applications:

→ Traversing
→ Browsing.

```
b) what are the key operations that can be performed
 on the stack are.
                               dequere (niaxien & E)
 → Kist → collections

→ Modules ← collections
                              The population tement is an
       stacks implementing lists.
List
       stack = [ ]
                         The star of the chark is: &
Ca:-
       stack. append (1)
       stack pop lement of the stack is 13. gop and
  print (len (stack) = = 0)
  print (Len (stack)) 7(3) oldining but tentine leas in
 collections module ) bassassi and dolder transfer and
       L'aequeue - double ended queue!
             class militable anteres has faille at
               -> appendely got sill-most and ad busha
               > POPL)
                                     example pregram.
 ex:-
                                      stock = [ ]
   import collections
   Stack = collections. deginene ()
                                       stark, apprond B)
   print (len(stack) -1)
                                      stack approve (8)
 from collections impost *
                                       stack, append (3)
  8 = dequeue (max len = 3)
                                       April (Stock[-1])
                     is" stack 1-21
                                   point (the tep element
  print(s)
                                         Carpag. Hooks
 # adding elements
                                        iff not stocker:
  3. append (12)
                     print ("The stack is empty")
  5. append (13)
                          point designation of house and it is
  5. append (14) Assistant Set Assis out to Atpast and I taken
  if len(s) == s.maxlen:
          print ("stack is ful")
  print ("The popped element is", s. pop())
                               that ade to affect said
  print ("The stack is", s)
  Print ("The size of the stack is", len(s))
  print ("The Top element is:", s[-1])
```

the war her ked teleduction that can be presioned Output :dequeue (maxlen = 3) Stack is full The popped element is 14 The stack is dequeue ([12,13], max len = 3) The size of the stack is: 2. The top element of the Stack 1s 13. 2) Last In first fout principle. (KIFO) (South) (1) The element which was inserted at first will go to the bottom of the stack and The Rast In elements will go to first, and to the detetion and adding

import cellections

Il adalng elements

H LEDICO == CHICKLE

print ("The stock is

thint ("The size of the stack is", sentes)

pint (" The rop element (s: " st-2)

(gr) bringing

pulat (Lentstack) -1)-

then collections inspost &

should be done from the top position. example program. stack = []

stack, append (1)

stack append (2)

stack.append(3)

Aprint (stack[-1]) print (the Top element is", stack [-1]) Stack.pop(@)

lif not stack (:

print ("The stack is empty")

print (Len(stack)), "The print (The length of the stack is (lencstack))

output: -

The Top element 1s: 3

The length of the stack is 2.

3) Discuss the principle of queue datastructure? Implement queue datastructure by using python remove the value pointed by Trant. program.

A) Queue 2 1 pd between al celular hart sar

It performs the Pirst in First out principle. (FIFO) The deletion and insertion can be done from the each end so that the first inserted element can be removed from the first or starting of the queue. Queues are the linear datastructures which are dynamic. The front and Rear positions will be the insertion and deletion parts points in queues.

paint ("Queue is maniphy

the trough of the queur is

Coucue, pop (6)

of print (den (queux))

- funtant

(Rear) insertion (-front) sular shary the cenqueue) was trived (dequeue)

Basic operations:-

- * Enqueue (insertion)
- * Dequeue (Deletion)
- print (the length of the queue is sentaneurs) unit ? !
- * Is Empty
- * Front (start element)
- * Rear (tod element)

a) enqueue (Insertion)

- -> check weather the queue is full or not.
- -> for the starting element, set the value of front to 3ero (0).
- -> Rear pointer is incremented by 1
- -> always insert the value at rear position.

b> Dequeue (deletion) -> check weather the queue is empty (or) not. -- Remove the value pointed by front. -> The front pointer is incremented by 1 for every dequeue operation. -> For the last element, the front and rear pointer reset ech and so that the first inserted element can example program: queue = [] Wild zonisuda aust con il adt on zona queue.append(1) queue. append (2) queue. append (3) queue. pop (0) if queue! The queue !" print (queue [0]) -> # peak value. if not queue: Bull operations: print ("Queue is mempty") # print (len (queue)) * Dequeur (Deletton) Quiput :print (The length of the queue is? len(queue)) Output: Classics teste) hard t

met wather the queue is the or not

The length of the queue is: 2.

s> What are the circular queues, now do they differ from regular queues?

A) The queues are the linear datastructures which are used to do the combination operations. It performs the first in first out principle and comming to circular queues these to are the extended versions of the linear queue. Ring buffering is possible in this circular queues. Once Rear reaches the last and index the insertion is not possible in the normal queue. To eversome this situation we use circular queue.

Basic operations:

1) enqueue

2) Dequeue

3) Is empty

A) Is full so Front

67 Rear.

working structure:

-front & rear =-1.

Enqueue :

Ocheck queue is full or not.

@ For start element set front value to Zerot O.

3 circularly increment the rear pointer by 1.

Assign the value at rear position.

Dequeue :-

Ocheck the queue is empty or not.

@ Remove the value pointed by front.

3 Front value is circularly incremented by 2.

@ The front & Rear pointer reset to -1.