Code Academy

Queues

Operations and applications

Code Academy

Lecture Goals

- Understand the meaning of **queue**.
- Know the **operations** and **application** of queue.
- Know the **implementation** of queues.
- Know the **advantages and disadvantages** of queues.

Code Academy

What is queue?

- Abstract data structure.
- It opens at both ends
- Can be implemented using different languages.
- The queue follows the FIFO (Fast in - First out) structure where the first element inserted would be the first element deleted.

Code Academy

The queue operation

- enqueue()
- dequeue()
- Peek()
- full()
- empty()

Code Academy

Insertion: enqueue

- The *enqueue* is an operation that inserts elements into the stack.
- The push algorithm:

  1) Start
  2) Add element to the queue (list).
  5) End

- To add an element to a queue (list) in python:
  use: list.append()

Code Academy

Deletion: dequeue()

- The *dequeue()* is a data manipulation operation which removes elements from the queue.
- The dequeue() algorithm:

  1) Start
  2) Remove element from the stack (list).
  5) End

- To remove an element from queue (list) in python use: list.pop()

Code Academy

Peek

- The *peak* is an operation retrieves the frontmost element within the stack, without deleting it.
- The peak algorithm:

  1) Start
  2) return element at the top of the stack.
  3) End

- To get the frontmost element in the stack using python use: list[-1]

Code Academy

full()

- The *full()* operation checks whether the queue is full. This operation is used to check the status of the stack with the help of top pointer.
- The full() algorithm:

  1) Start
  2) If the size of the queue is equal to the front position of the queue, the queue is full. Return 1.
  3) otherwise, return 0.
  4) End

Code Academy

empty()

- The *empty()* operation verifies whether the stack is empty. This operation is used to check the status of the stack with the help of top pointer.
- The empty() algorithm:

1) Start
2) If the front value is -1, the queue is empty. Return 1.
3) otherwise, return 0.
4) End

Code Academy

Queues Application

- CPU scheduling
- Routers and switches in networking.
- Handling website traffic.
- Memory management.

Code Academy

CPU scheduling

- Operating System has to define which process the CPU will be given. In Multiprogramming systems, the Operating system schedules the processes on the CPU to have the maximum utilization of it.
- One method is to use a queue to schedule.

Code Academy

Implementation

How to implement queue in python?

Code Academy

Implementation

- **List**
- Collections.deque

Code Academy

enqueue implementation

```
# Initialize the queue as a list
queue = []

# Adding elements to the queue
queue.append('a')
queue.append('b')
queue.append('c')

print("Initial queue")
print(queue)
```

```
Initial queue
['a', 'b', 'c']
```

Code Academy

dequeue implementation

```
# Removing elements from the queue
print("Elements dequeued from queue")
print(queue.pop(0))
print(queue.pop(0))
print(queue.pop(0))

print("\nQueue after removing elements")
print(queue)
```

```
Elements dequeued from queue
a
b
c

Queue after removing elements
[]
```

Code Academy

Advantages of queues

- A large amount of data can be managed efficiently with ease.
- Operation such as insertion and deletion can be done with ease.
- Queues are useful when a particular service is used by multiple consumer.

Code Academy

Disadvantages of queues

- Operation of insertion and deletion from the middle are time consuming.
- Limited space.