ININ1Y.UZ

PYTHONDA QUEUE

Biz Bilan 0 dan Cheksizlikkacha

CREATED BY: ALIMARDON BOQIJONOV

DARS TARTIBI

QUEUE

INFIN1TY.UZ

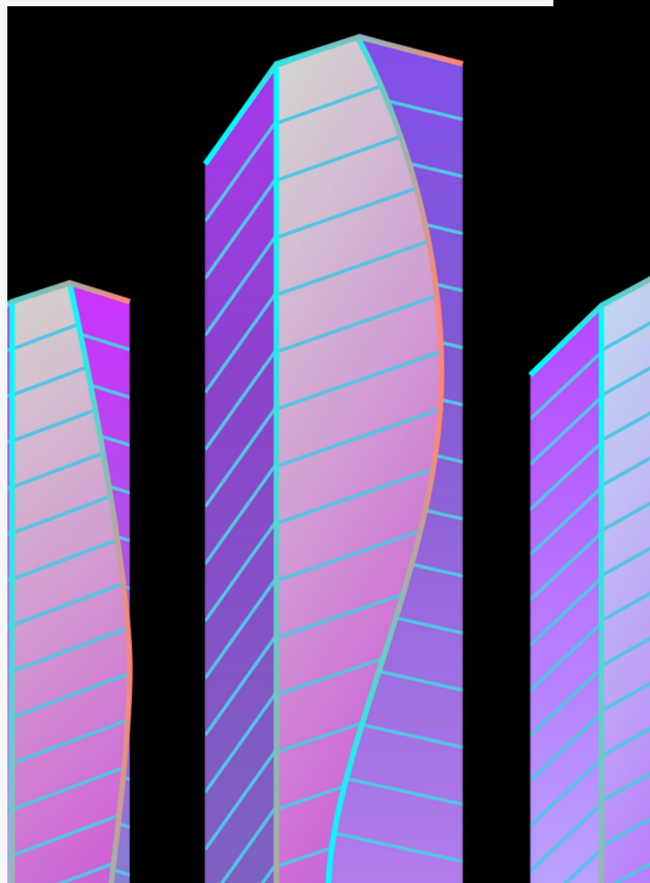
Queue dasturiy yordamchi tuzilmalardan biridir. Ushbu tuzilma elementlarni saqlash va ularga qo'shish va o'chirishni taminlaydi. Queue tuzilmasi asosan birinchi kirgan birinchi chiqariladi (First-In-First-Out, FIFO) tartibda ishlaydi. Ya'ni, elementlar birinchi kiritilgan tartibda saqlanadi va birinchi kiritilgan element birinchi bo'lib olinadi.

Queue tuzilmasi umumiyat bilan har xil dasturlash tillarida va platformalarda foydalaniladi. Misol uchun, ma'lumotlarni qayta ishlashning zarur bo'lishi, veb-saytlardagi so'rov qatorlarini to'plash, tarmoq protokollarida paketlar ustida amalga oshirish, barcha turdagi tizimlarda to'plamalar qatorini boshqarish va boshqalar kabi ko'plab ishlarda qo'llaniladi.

Queue tuzilmasi odatda `**enqueue**` va `**dequeue**` amallari bilan bajariladi. `**Enqueue**` yordamida elementlar queue'ga qo'shiladi, va `**dequeue**` yordamida esa birinchi kirgan element olib tashlanadi. Queue'da elementlar ro'yxatda saqlanadi va ularga tartib bilan murojaat qilish mumkin.

Queue, foydalanish mumkin bo'lgan ikki asosiy amalni o'z ichiga oladi:

1. Enqueue: Ma'lumotni qatorning oxiriga qo'shish.
2. Dequeue: Ma'lumotni qatorning boshidan olib tashlash.



Queue qator (line) deb tasavvur qilinadi, ma'lumotlar qatorning oxiridan qo'shiladi va boshidan chiqariladi. Bu struktura ma'lumotlarni o'z tartibida saqlayishda va ustunlik bilan ma'lumotlar saralash, jarayonlar yo'li, tasklar yoki boshqa ishlar uchun foydalaniladi.

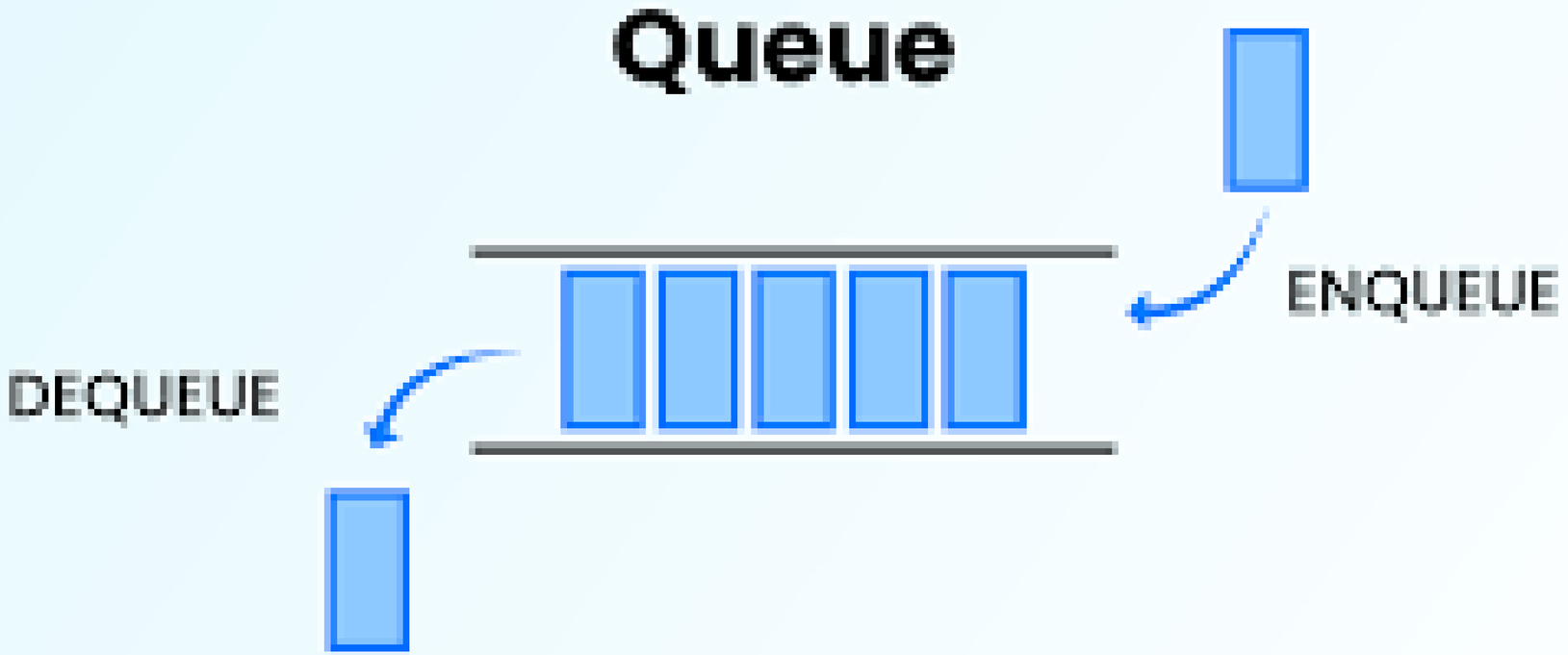
Masalan, internetga muvofiq saytlarga murojaatlar turli tartibda qatorga qo'shiladi va serverning mavjud resurslarini foydalanib qaytarish uchun qatorning boshidan har bir murojaatni bajaradi.

(1) queue

masai

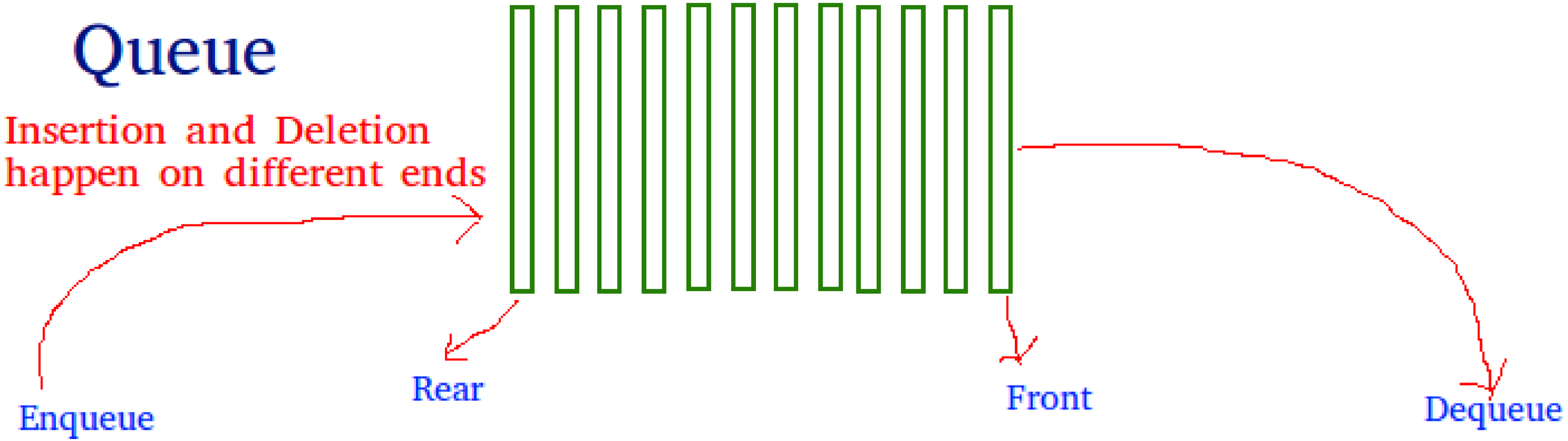
QUEUE
DATA STRUCTURE

(2) queue-navbat



Queue

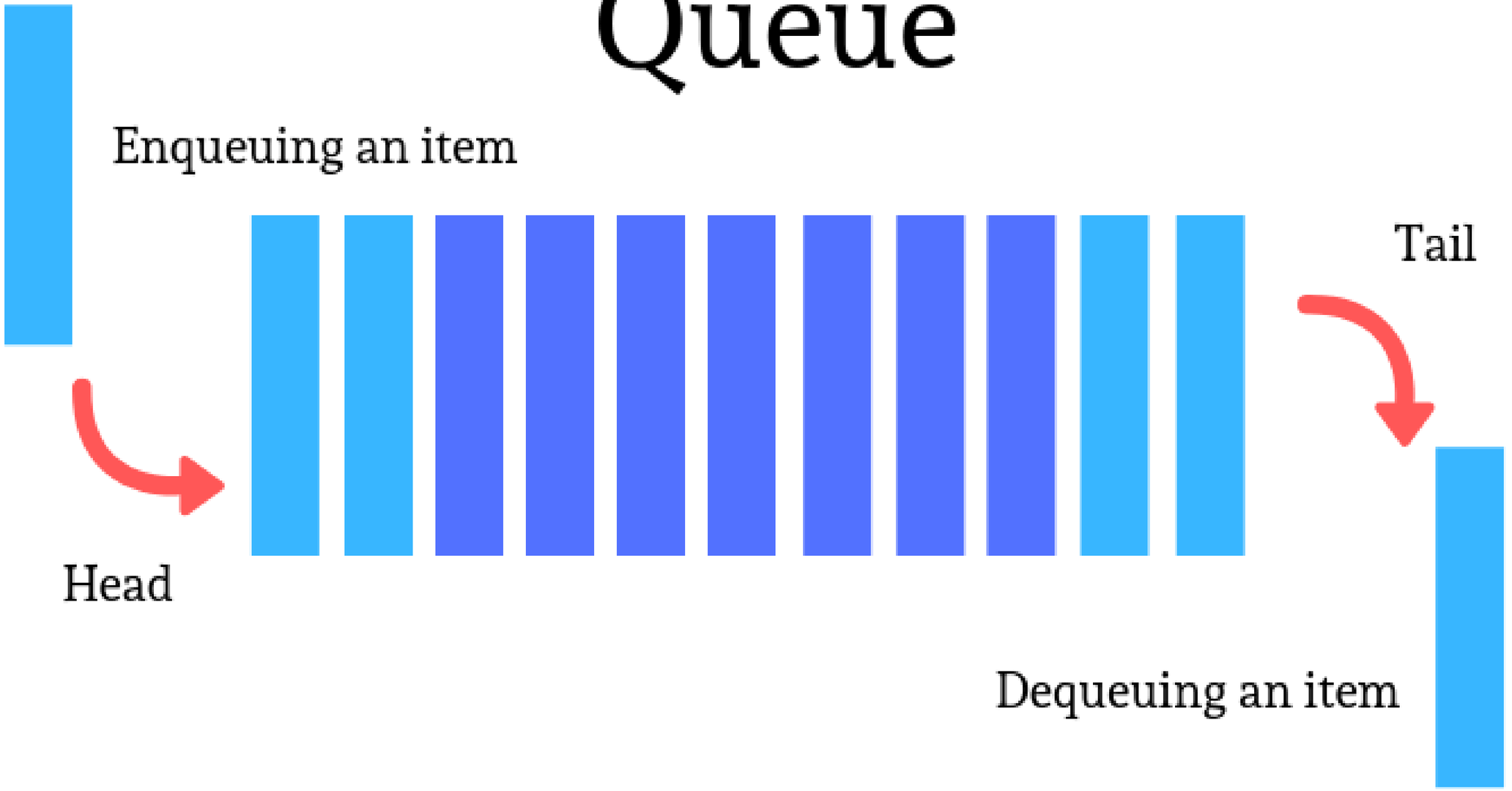
Insertion and Deletion happen on different ends



First in, first out

Queue

learnersbucket.com



python Copy code

```python
import queue

# Bo'sh quyuhiyatli navbat yaratish
pq = queue.PriorityQueue()

# Elementlarni quyuhiyatli navbatga qo'shish
pq.put((3, 'Olma'))  # (prioritet, element)
pq.put((1, 'Banana'))
pq.put((2, 'Sher'))
pq.put((5, 'Durian'))
pq.put((4, 'Anor'))

# Elementlarni quyuhiyatli navbatdan olib tashlash
while not pq.empty():
    element = pq.get()
    print(element[1])  # Element qiymatini chiqarish

# Natija:
# Banana
# Sher
# Olma
# Anor
# Durian
```

Yuoqridagi misolda, PriorityQueue obyekti pq yaratiladi va put() metodidan foydalanib elementlar qo'shiladi. Har bir element (prioritet, element) tuple sifatida ifodalangan, prioritet elementning prioritetini ifodalaydi. Kichik qiymatlar yuqori prioriteta ishora qiladi. Keyin, elementlarni quyuhiyatli navbatdan olib tashlash uchun get() metodi ishlatiladi, bu metodi yuqori prioritetli elementni qaytaradi. Quyuhiyatli navbatdagi elementlar avtomatik ravishda prioritetalarga ko'ra saralandi, shuning uchun ularni to'g'ri tartibda olishimiz mumkin.

Yuoqridagi misolda, `PriorityQueue` obyekti `pq` yaratiladi va `put()` metodidan foydalanib elementlar qo'shiladi. Har bir element `(prioritet, element)` tuple sifatida ifodalangan, `prioritet` elementning prioritetini ifodalaydi. Kichik qiymatlar yuqori prioriteta ishora qiladi. Keyin, elementlarni quyuhiyatli navbatdan olib tashlash uchun `get()` metodi ishlatiladi, bu metodi yuqori prioritetli elementni qaytaradi.

Quyuhiyatli navbatdagi elementlar avtomatik ravishda prioritetalarga ko'ra saralandi, shuning uchun ularni to'g'ri tartibda olishimiz mumkin.

Yuoqridagi misolda `CircularQueue` nomli sinf yaratiladi. `__init__` metodida navbat olchami (`k`), ro'yxat (`queue`) va boshlang'ich indekslar (`front` va `rear`) aniqlanadi. `is_empty` va `is_full` metodlar orqali navbatning bo'shligini va to'la bo'lganligini tekshirish mumkin. `enqueue` metodida yangi element navbatga qo'shiladi, va `dequeue` metodida esa eng oldingi element o'chiriladi. Circular Queue da "front" va "rear" indekslari navbatni qanday aylantirishni ifodalaydi. Agar navbat to'la bo'lsa, yangi element qo'shilmaydi. Agar navbat bo'sh bo'lsa, `front` va `rear` indekslari birinchi elementga birgalikda bo'ladigan indekslarga o'zgartiriladi. Navbat to'la bo'lganda va bo'sh bo'lganda xatolik xabarlari chiqariladi. Elementlar navbatdan olib tashlanganda ular qiymatlarini chiqarish mumkin.

Misol kodimizda 5 elementli Circular Queue yaratiladi. Elementlar qo'shiladi va olib tashlanadi. Agar navbat to'la bo'lsa yoki bo'sh bo'lsa, xatolik xabarlari chiqariladi. Elementlar olib tashlanganda esa ularning qiymatlari konsolga chiqariladi.

Duble-ended Queue

Yuoqridagi misolda `Deque` nomli sinf yaratiladi. `__init__` metodida deque ro'yxati (`deque`) yaratiladi. `is_empty` metodida deque bo'shligini tekshirish mumkin. `add_front` metodi deque boshidan element qo'shishni, `add_rear` metodi esa ohiridan element qo'shishni amalga oshiradi. `remove_front` va `remove_rear` metodlar orqali esa deque elementlarini o'chirish mumkin. `size` metodi deque hajmini qaytaradi.

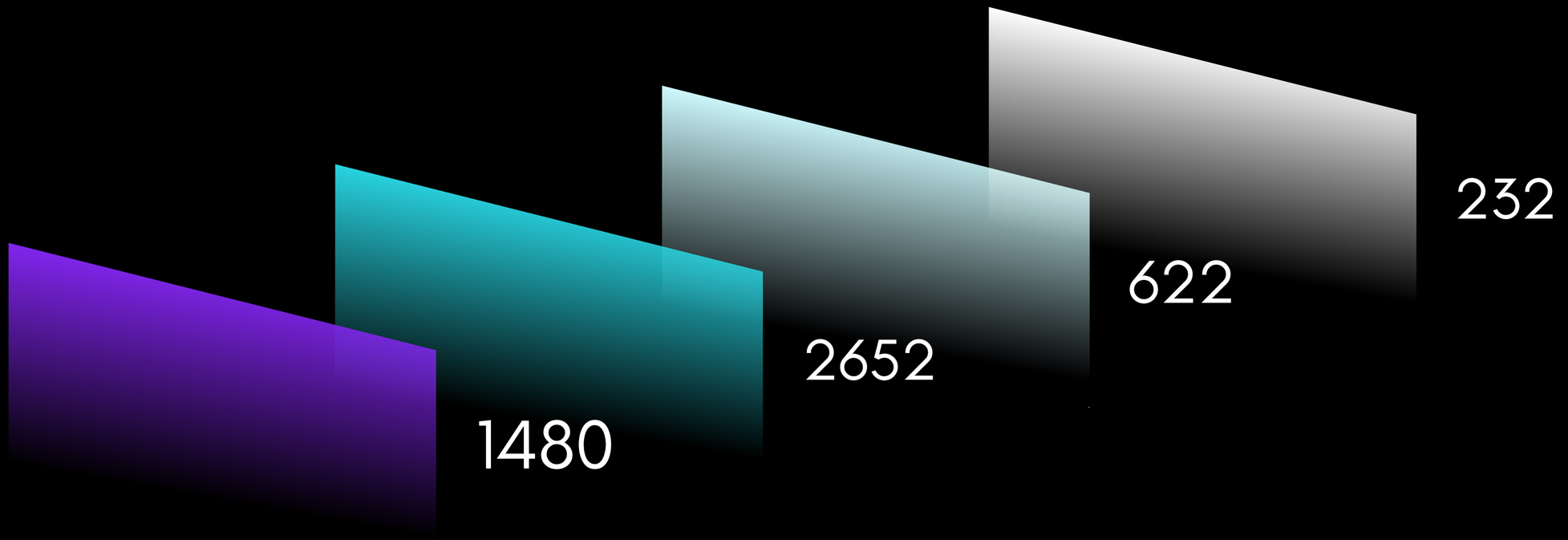
Misol kodimizda `Deque` obyekti yaratiladi. Elementlar boshidan va ohiridan qo'shiladi va o'chiriladi. Deque hajmi konsolga chiqariladi.

LEETCODE

INFIN1TY.UZ

BACK TO NAVIGATION PAGE

Array va Queuega oid masala ishlash



THANK YOU FOR YOUR ATTENTION

INFIN1TY.UZ