**LAPORAN PRAKTIKUM**

**STRUKTUR DATA**

**MODUL KE-4**

**QUEUE PADA PYTHON**



**Disusun Oleh:**

**Nama :** Oktario Mufti Yudha

**NPM :** 2320506044

**Kelas :** 04 (Empat)

**Program Studi S1 Teknologi Informasi**

**Fakultas Teknik, Universitas Tidar**

**Genap 2023/2024**

1. **Tujuan Praktikum**

Praktikum ini bertujuan untuk memperkenalkan mahasiswa pada konsep dasar Queue serta memberikan pemahaman praktis dalam mengimplementasikannya menggunakan bahasa pemrograman Python. Praktikum ini bertujuan untuk memberikan mahasiswa pemahaman yang kuat tentang bagaimana Queue beroperasi, termasuk konsep enqueue (menambahkan elemen), dequeue (menghapus elemen), dan peek (melihat elemen paling awal tanpa menghapusnya).

1. **Dasar Teori**

struktur data queue adalah konsep FIFO (First In, First Out), yang berarti elemen pertama yang dimasukkan ke dalam tumpukan adalah elemen pertama yang akan dikeluarkan. Struktur data ini mirip dengan antrian pada rumah sakit, dimana pasien yang pertama mendaftar adalah pasien yang akan dipanggil pertama kali.

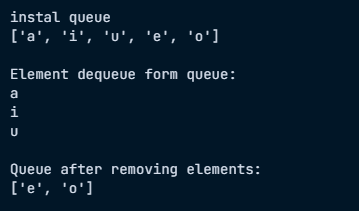
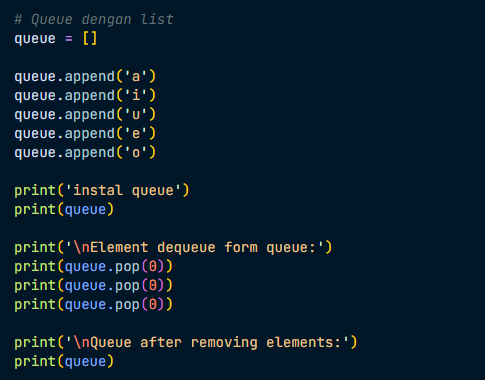
Operasi dasar yang biasa digunakan adalah:

1. Enqueue (Menambahkan elemen): Operasi untuk menambahkan elemen ke dalam antrian. Elemen baru akan ditambahkan di ujung rear/tail dari antrian.

2. Dequeue (Menghapus elemen): Operasi untuk menghapus elemen dari antrian. Elemen yang dihapus adalah elemen yang berada di ujung front/head dari antrian.

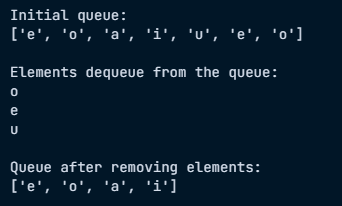
3. Peek (Melihat elemen teratas): Operasi untuk melihat elemen teratas dalam antrian tanpa menghapusnya. Ini memungkinkan untuk memeriksa elemen yang berikutnya akan dihapus.

1. **Hasil dan Pembahasan**
   1. Queue dengan List

****

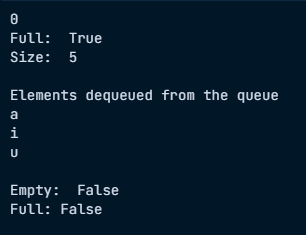
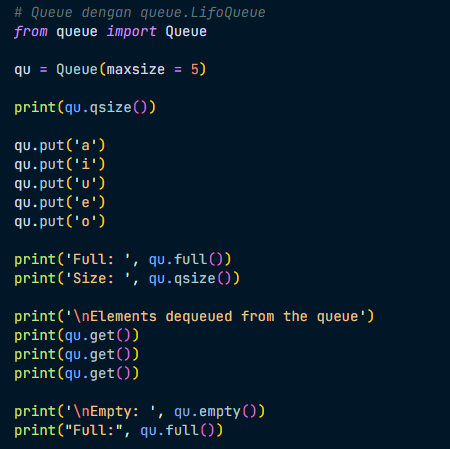
*Gambar 3.1: Queue dengan List dan Outputnya*

1. queue = []: Membuat sebuah list kosong yang akan digunakan untuk menyimpan elemen-elemen dalam antrian.
2. queue.append('a'): Huruf 'a' ditambahkan ke dalam antrian menggunakan metode .append().
3. queue.append('i'): Huruf 'i' ditambahkan ke dalam antrian, menjadi elemen kedua dalam antrian.
4. queue.append('u'): Huruf 'u' ditambahkan ke dalam antrian, menjadi elemen ketiga dalam antrian.
5. queue.append('e'): Huruf 'e' ditambahkan ke dalam antrian, menjadi elemen keempat dalam antrian.
6. queue.append('o'): Huruf 'o' ditambahkan ke dalam antrian, menjadi elemen kelima dan terakhir dalam antrian.
7. print('instal queue'): Mencetak string 'instal queue'.
8. print(queue): Mencetak isi dari antrian menggunakan fungsi print().
9. print('\nElement dequeue form queue:'): Mencetak string '\nElement dequeue form queue:'.
10. print(queue.pop(0)): Menghapus dan mencetak elemen pertama dari antrian menggunakan metode .pop(0). Huruf 'a' dihapus dari antrian dan dicetak.
11. print(queue.pop(0)): Menghapus dan mencetak elemen kedua dari antrian. Huruf 'i' dihapus dari antrian dan dicetak.
12. print(queue.pop(0)): Menghapus dan mencetak elemen ketiga dari antrian. Huruf 'u' dihapus dari antrian dan dicetak.
13. print('\nQueue after removing elements:'): Mencetak string '\nQueue after removing elements:'.
14. print(queue): Mencetak isi dari antrian setelah tiga elemen pertamanya dihapus.
15. Queue dengan collections.deque



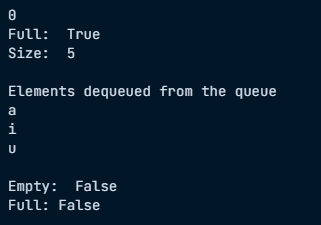
*Gambar 3.2: Queue dengan collections.deque dan Outputnya*

1. from collections import deque : Mengimpor kelas deque dari modul collections.
2. queue.append('a'), queue.append('i'), queue.append('u'), queue.append('e'), queue.append('o'): Huruf ‘a, i, u, e, o’ ditambahkan ke dalam antrian menggunakan metode .append().
3. print('Initial queue:'): Mencetak string 'Initial queue:'.
4. print(queue): Mencetak isi dari antrian menggunakan fungsi print().
5. print('\nElements dequeue from the queue:'): Mencetak string '\nElements dequeue from the queue:'.
6. print(queue.pop()), print(queue.pop()), print(queue.pop()): Menghapus dan mencetak elemen-elemen dari antrian menggunakan metode .pop().
7. print('\nQueue after removing elements:'): Mencetak string '\nQueue after removing elements:'.
8. print(queue): Mencetak isi dari antrian setelah tiga elemen terakhirnya dihapus.
9. Queue dengan queue.LifoQueue



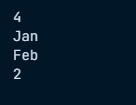
*Gambar 3.3: Queue dengan queue.LifoQueue dan output*

1. from queue import Queue: Mengimpor kelas Queue dari modul queue.
2. qu = Queue(maxsize = 5): Membuat objek antrian qu dengan ukuran maksimum (maxsize) sebesar 5.
3. print(qu.qsize()): Mencetak jumlah elemen saat ini dalam antrian menggunakan metode .qsize().
4. qu.put('a'), qu.put('i'), qu.put('u'), qu.put('e'), qu.put('o'): Menambahkan huruf vokal ke dalam antrian menggunakan metode .put().
5. print('Full: ', qu.full()): Mencetak apakah antrian sudah penuh menggunakan metode .full(), yang akan mengembalikan True jika antrian sudah penuh.
6. print('Size: ', qu.qsize()): Mencetak jumlah elemen saat ini dalam antrian menggunakan metode .qsize().
7. print('\nElements dequeued from the queue'): Mencetak string '\nElements dequeued from the queue'.
8. print(qu.get()), print(qu.get()), print(qu.get()): Menghapus dan mencetak elemen-elemen dari antrian menggunakan metode .get().
9. print('\nEmpty: ', qu.empty()): Mencetak apakah antrian kosong menggunakan metode .empty(), yang akan mengembalikan True jika antrian kosong.
10. print("Full:", qu.full()): Mencetak apakah antrian penuh menggunakan metode .full().
11. Queue dengan queue.Queue

*Gambr 3.4:* Queue dengan queue.Queue

1. from queue import Queue: Mengimpor kelas Queue dari modul queue.
2. qu = Queue(maxsize = 5): Baris ini membuat objek antrian qu dengan ukuran maksimum (maxsize) sebesar 5.
3. print(qu.qsize()): Mencetak jumlah elemen saat ini dalam antrian menggunakan metode .qsize().
4. qu.put('a'), qu.put('i'), qu.put('u'), qu.put('e'), qu.put('o'): Menambahkan huruf vokal ke dalam antrian menggunakan metode .put().
5. print('Full: ', qu.full()): Mencetak apakah antrian sudah penuh menggunakan metode .full(), yang akan mengembalikan True jika antrian sudah penuh.
6. print('Size: ', qu.qsize()): Mencetak jumlah elemen saat ini dalam antrian menggunakan metode .qsize().
7. print('\nElements dequeued from the queue'): Mencetak string '\nElements dequeued from the queue'.
8. print(qu.get()), print(qu.get()), print(qu.get()): Menghapus dan mencetak elemen-elemen dari antrian menggunakan metode .get().
9. print('\nEmpty: ', qu.empty()): Mencetak apakah antrian kosong menggunakan metode .empty().
10. print("Full:", qu.full()): Mencetak apakah antrian penuh menggunakan metode .full().
11. Queue dengan Singel Linked List

*Gambr 3.5:* Queue dengan Singel Linked List

1. class Queue:: Mendefinisikan sebuah kelas Python bernama Queue.
2. def \_\_init\_\_(self):: Metode konstruktor kelas Queue, yang akan dipanggil ketika objek dari kelas ini dibuat.
3. self.queue = list(): Membuat list kosong yang akan digunakan untuk menyimpan elemen-elemen antrian.
4. def addtoqu(self, dataval):: Mendefinisikan metode addtoqu() untuk menambahkan elemen baru ke dalam antrian.
5. if dataval not in self.queue:: Memeriksa apakah dataval belum ada dalam antrian.
6. self.queue.insert(0, dataval): Jika dataval belum ada dalam antrian, metode ini akan menambahkannya ke posisi awal antrian menggunakan metode .insert().
7. return True: Mengembalikan True setelah menambahkan dataval ke dalam antrian.
8. return False: Mengembalikan False jika dataval sudah ada dalam antrian.
9. def removefromqu(self):: Mendefinisikan metode removefromqu() untuk menghapus elemen dari antrian.
10. if len(self.queue) > 0:: Memeriksa apakah antrian tidak kosong.
11. return self.queue.pop(): Menghapus dan mengembalikan elemen terakhir dari antrian menggunakan metode .pop().
12. return ("No elements in Queue!"): Mengembalikan pesan "No elements in Queue!" jika antrian kosong.
13. def size(self):: Mendefinisikan metode size() untuk mengembalikan jumlah elemen dalam antrian.
14. return len(self.queue): Mengembalikan panjang (jumlah elemen) dari antrian menggunakan fungsi len().
15. TheQueue = Queue(): Membuat sebuah objek TheQueue dari kelas Queue.
16. TheQueue.addtoqu("Jan"), TheQueue.addtoqu("Feb"), TheQueue.addtoqu("March"), TheQueue.addtoqu("April"): Menambahkan beberapa elemen ke dalam antrian menggunakan metode addtoqu().
17. print(TheQueue.size()): Mencetak jumlah elemen dalam antrian menggunakan metode size().
18. print(TheQueue.removefromqu()), print(TheQueue.removefromqu()): Menghapus dan mencetak elemen-elemen dari antrian menggunakan metode removefromqu().
19. print(TheQueue.size()): Mencetak jumlah elemen dalam antrian setelah beberapa elemen dihapus.
20. **Kesimpulan**

Praktikum stack membantu kita memahami konsep dasar Queue dalam pemrograman. Kita belajar tentang cara menambahkan dan menghapus elemen pada Queue, serta bagaimana tumpukan bekerja menggunakan aturan FIFO (First In, First Out). Kita dapat melihat bagaimana Queue digunakan dalam pemrograman sehari-hari. Praktikum ini memberikan pemahaman yang kuat tentang struktur data ini dan bagaimana kita bisa menggunakan mereka dalam memecahkan masalah komputasi.