



ALGORITMA DAN STRUKTUR DATA

INF1083

LAPORAN PRAKTIKUM 7 : Struktur Data Dasar (2)

Oleh :

Akhmad Qasim

2211102441237

Teknik Informatika
Fakultas Sains & Teknologi
Universitas Muhammadiyah Kalimantan Timur

Samarinda, 2023

Laporan Praktikum 7:

Struktur Data Dasar (2)

Pokok Bahasan:

- ❖ Queue.
- ❖ Deque.
- ❖ Unordered List.
- ❖ Ordered List.

Tujuan Pembelajaran:

- ✓ Memahami implementasi *Queue* pada struktur data *Python*.
- ✓ Memahami implementasi *Deque* pada struktur data *Python*.
- ✓ Memahami implementasi *Unordered List* pada struktur data *Python*.
- ✓ Memahami implementasi *Ordered List* pada struktur pada *Python*.

Percobaan & Latihan:

1. Queue

a) Berikan tampilan output dari perintah diatas!

Tampilan Output:

```
3
False
None
4
dog
2
```

b) Jelaskan fungsi baris kode 17 hingga 29!

```
18 q = Queue() # Membuat objek Queue
19
20 q.enqueue(4) # Memasukkan nilai 4 ke dalam queue
21 q.enqueue('dog') # Memasukkan nilai 'dog' ke dalam queue
22 q.enqueue(True) # Memasukkan nilai True ke dalam queue
23
24 print(q.size()) # Menampilkan berapa banyak item dari queue
25 print(q.isEmpty()) # Menampilkan apakah queue kosong atau tidak
26 print(q.enqueue(8.4)) # Memasukkan nilai 8.4 ke dalam queue
27 print(q.dequeue()) # Menghapus nilai dari queue dan menampilkan nilai yang dihapus
28 print(q.dequeue()) # Menghapus nilai dari queue dan menampilkan nilai yang dihapus
29 print(q.size()) # Menampilkan berapa banyak item dari queue
```

2. Queue

a) Berikan tampilan output dari perintah diatas!

Tampilan Output:

```
Susan
```

b) Ubahlah nilai 7 pada baris 16 (print(...,7)) dengan nilai 6, kemudian berikan tampilan dan berikan analisisnya!

Tampilan Output:

```
Kent
```

Pada Percobaan & Latihan 7.2 kita menggunakan algoritma Hot Potato yang memproses antrian dengan memutar elemen-elemennya dengan nomor yang diberikan dan menghapus elemen setiap iterasi selesai. Pada contoh kode di atas, algoritma Hot Potato dimulai dengan sebuah list yang berisi nama-nama ("Bill", "David", "Susan", "Jane", "Kent", "Brad"). Kemudian, setiap nama akan dimasukkan ke dalam antrian yang disimpan dalam variabel simqueue. Pada saat iterasi pertama, nilai num adalah 6, sehingga antrian akan diputar sebanyak 6 kali, sehingga elemen pada posisi ke-6 akan dikeluarkan dari antrian. Karena elemen pada posisi ke-6 adalah "Kent", maka output dari kode di atas adalah "Kent". Namun jika num diganti menjadi 7, maka output dari kode di atas adalah "Susan", karena elemen pada posisi ke-7 dari antrian adalah "Susan". Hal ini karena algoritma Hot Potato memutar antrian sebanyak 7 kali sehingga elemen pada posisi ke-7 akan dikeluarkan.

3. Queue

a) Berikan tampilan output dari perintah diatas dan hasil analisa!

Tampilan Output:

```
Average Wait 65.13 secs 0 tasks remaining.  
Average Wait 180.80 secs 3 tasks remaining.  
Average Wait 786.65 secs 4 tasks remaining.  
Average Wait 80.62 secs 4 tasks remaining.  
Average Wait 107.42 secs 4 tasks remaining.  
Average Wait 110.47 secs 2 tasks remaining.  
Average Wait 66.41 secs 0 tasks remaining.  
Average Wait 175.41 secs 0 tasks remaining.  
Average Wait 44.00 secs 0 tasks remaining.  
Average Wait 133.06 secs 0 tasks remaining.
```

Hasil analisa:

```

1  from pythonds.basic.queue import Queue # Mengimpor modul Queue
2  import random # Mengimpor modul random
3
4
5  ─ Akhmad Qasim *
6  class Printer: # Membuat kelas Printer
7      new *
8      def __init__(self, ppm): # Membuat konstruktor
9          self.pagerate = ppm # Membuat variabel pagerate dengan nilai ppm
10         self.currentTask = None # Membuat variabel currentTask dengan nilai None
11         self.timeRemaining = 0 # Membuat variabel timeRemaining dengan nilai 0
12
13     new *
14     def tick(self): # Membuat fungsi tick
15         if self.currentTask != None: # Jika currentTask tidak sama dengan None
16             self.timeRemaining = self.timeRemaining - 1 # Maka nilai timeRemaining akan dikurangi 1
17             if self.timeRemaining <= 0: # Jika nilai timeRemaining kurang dari sama dengan 0
18                 self.currentTask = None # Maka nilai currentTask akan bernilai None
19
20     new *
21     def busy(self): # Membuat fungsi busy
22         if self.currentTask != None: # Jika currentTask tidak sama dengan None
23             return True # Maka kembalikan nilai True
24         else: # Maka jika tidak
25             return False # Maka kembalikan nilai False
26
27
28  ─ Akhmad Qasim *
29  def startNext(self, newtask): # Membuat fungsi startNext
30      self.currentTask = newtask # Membuat variabel currentTask dengan nilai newtask
31      # Membuat variabel timeRemaining dengan nilai newtask.getPages() * 60 / self.pagerate
32      self.timeRemaining = newtask.getPages() * 60 / self.pagerate
33
34  ─ Akhmad Qasim *
35  class Task: # Membuat kelas Task untuk membuat task baru
36      new *
37      def __init__(self, time): # Membuat konstruktor
38          self.timestamp = time # Membuat variabel timestamp dengan nilai time
39          self.pages = random.randrange(1, 21) # Membuat variabel pages dengan nilai random dari 1 sampai 21
40
41      new *
42      def getStamp(self): # Membuat fungsi getStamp
43          return self.timestamp # Mengembalikan nilai timestamp
44
45      new *
46      def getPages(self): # Membuat fungsi getPages
47          return self.pages # Mengembalikan nilai pages
48
49      new *
50      def waitTime(self, currenttime): # Membuat fungsi waitTime
51          return currenttime - self.timestamp # Mengembalikan nilai currenttime - timestamp
52
53
54  ─ Akhmad Qasim *
55  def simulation(numSeconds, pagesPerMinute): # Membuat fungsi simulation
56      labprinter = Printer(pagesPerMinute) # Membuat objek labprinter dengan nilai Printer(pagesPerMinute)
57      printQueue = Queue() # Membuat objek printQueue dengan nilai Queue()
58      waitingtimes = [] # Membuat variabel waitingtimes dengan nilai list kosong
59
60      for currentSecond in range(numSeconds): # Melakukan perulangan for
61          if newPrintTask(): # Jika fungsi newPrintTask bernilai True
62              task = Task(currentSecond) # Membuat objek task dengan nilai Task(currentSecond)
63              printQueue.enqueue(task) # Memasukkan nilai task ke dalam printQueue
64
65          # Jika labprinter tidak sibuk dan printQueue tidak kosong
66          if (not labprinter.busy()) and (not printQueue.isEmpty()):
67              nexttask = printQueue.dequeue() # Membuat objek nexttask dengan nilai printQueue.dequeue()
68              # Menambahkan nilai nexttask.waitTime(currentSecond) ke dalam waitingtimes
69              waitingtimes.append(nexttask.waitTime(currentSecond))
70              labprinter.startNext(nexttask) # Memulai task selanjutnya
71
72      labprinter.tick() # Memanggil fungsi tick

```

b) Sebutkan rata-rata waktu terlama dan tercepat dari hasil ouput!

Waktu terlama: 786 Detik

Waktu tercepat: 44 Detik

4. Deque

a) Berikan tampilan output dari perintah diatas!

Tampilan output:

```
True
4
False
8.4
True
```

b) Jelaskan fungsi baris kode 23 hingga 33!

Tampilan syntax:

```
24 d = Deque() # Membuat objek d dengan nilai Deque()
25 print(d.isEmpty()) # Menampilkan apakah d kosong
26 d.addRear(4) # Menambahkan nilai 4 ke dalam d
27 d.addRear('dog') # Menambahkan nilai 'dog' ke dalam d
28 d.addFront('cat') # Menambahkan nilai 'cat' ke dalam d
29 d.addFront(True) # Menambahkan nilai True ke dalam d
30 print(d.size()) # Menampilkan panjang dari d
31 print(d.isEmpty()) # Menampilkan apakah d kosong
32 d.addRear(8.4) # Menambahkan nilai 8.4 ke dalam d
33 print(d.removeRear()) # Menampilkan nilai yang dihapus dari d
34 print(d.removeFront()) # Menampilkan nilai yang dihapus dari d
```

5. Deque

a) Berikan hasil ouput dan analisa!

Tampilan output:

```
False
True
```

Tampilan syntax:

```

1 from pythonds.basic.deque import Deque # Mengimpor modul Deque
2
3
4 def palchecker(aString): # Membuat fungsi palchecker
5     chardeque = Deque() # Membuat objek chardeque dengan nilai Deque()
6
7     for ch in aString: # Melakukan perulangan untuk setiap nilai ch dalam aString
8         chardeque.addRear(ch) # Menambahkan nilai ch ke dalam chardeque
9
10    stillEqual = True # Membuat variabel stillEqual dengan nilai True
11
12    # Melakukan perulangan selama chardeque lebih besar dari 1 dan stillEqual bernilai True
13    while chardeque.size() > 1 and stillEqual:
14        # Menghapus nilai pertama dari chardeque dan menyimpannya ke dalam variabel first
15        first = chardeque.removeFront()
16        # Menghapus nilai terakhir dari chardeque dan menyimpannya ke dalam variabel last
17        last = chardeque.removeRear()
18        if first != last: # Jika first tidak sama dengan last
19            stillEqual = False # Maka nilai stillEqual akan bernilai False
20
21    return stillEqual # Mengembalikan nilai stillEqual
22
23
24 print(palchecker("lsdkjfskf")) # Mencetak hasil dari palchecker("lsdkjfskf")
25 print(palchecker("radar")) # Mencetak hasil dari palchecker("radar")

```

6. Unordered

a) Berikan tampilan output dari perintah diatas!

Tampilan output:

```

6
True
False
True
7
6
5
4
False

```

b) Jelaskan fungsi baris kode 68 hingga 91!

Tampilan syntax:

```
68 mylist = UnorderedList() # Membuat objek mylist dengan nilai UnorderedList()
69
70 mylist.add(31) # Menambahkan nilai 31 ke dalam mylist
71 mylist.add(77) # Menambahkan nilai 77 ke dalam mylist
72 mylist.add(17) # Menambahkan nilai 17 ke dalam mylist
73 mylist.add(93) # Menambahkan nilai 93 ke dalam mylist
74 mylist.add(26) # Menambahkan nilai 26 ke dalam mylist
75 mylist.add(54) # Menambahkan nilai 54 ke dalam mylist
76
77 print(mylist.size()) # Menampilkan panjang dari mylist
78 print(mylist.search(93)) # Menampilkan apakah 93 ada di dalam mylist
79 print(mylist.search(100)) # Menampilkan apakah 100 ada di dalam mylist
80
81 mylist.add(100) # Menambahkan nilai 100 ke dalam mylist
82 print(mylist.search(100)) # Menampilkan apakah 100 ada di dalam mylist
83 print(mylist.size()) # Menampilkan panjang dari mylist
84
85 mylist.remove(54) # Menghapus nilai 54 dari mylist
86 print(mylist.size()) # Menampilkan panjang dari mylist
87 mylist.remove(93) # Menghapus nilai 93 dari mylist
88 print(mylist.size()) # Menampilkan panjang dari mylist
89 mylist.remove(31) # Menghapus nilai 31 dari mylist
90 print(mylist.size()) # Menampilkan panjang dari mylist
91 print(mylist.search(93)) # Menampilkan apakah 93 ada di dalam mylist
```

7. Ordered

a) Berikan tampilan output dari perintah diatas!

Tampilan output:

```
6
True
False
```

b) Jelaskan fungsi baris kode 70 hingga 80!

Tampilan syntax:

```
70 mylist = OrderedDict() # Membuat objek mylist dengan nilai OrderedDict()
71 mylist.add(31) # Menambahkan nilai 31 ke dalam mylist
72 mylist.add(77) # Menambahkan nilai 77 ke dalam mylist
73 mylist.add(17) # Menambahkan nilai 17 ke dalam mylist
74 mylist.add(93) # Menambahkan nilai 93 ke dalam mylist
75 mylist.add(26) # Menambahkan nilai 26 ke dalam mylist
76 mylist.add(54) # Menambahkan nilai 54 ke dalam mylist
77
78 print(mylist.size()) # Menampilkan panjang dari mylist
79 print(mylist.search(93)) # Menampilkan apakah nilai 93 ada di dalam mylist
80 print(mylist.search(100)) # Menampilkan apakah nilai 100 ada di dalam mylist
```

Kesimpulan:

Queue adalah struktur data dengan aturan first in, first out (FIFO) yang digunakan dalam penjadwalan tugas, simulasi antrian, dan pemrosesan dokumen. Deque adalah struktur data yang mirip dengan queue namun dapat menambahkan dan menghapus elemen di kedua ujung (depan dan belakang) dan dapat digunakan untuk implementasi antrian prioritas. Unordered List adalah jenis list dengan elemen yang tidak memiliki urutan tertentu dan dilakukan menggunakan linked list atau array dengan operasi dasar seperti append, remove, search, dan length. Ordered List adalah jenis list dengan elemen yang disusun secara teratur dengan implementasi linked list atau array yang memastikan bahwa elemen teratur secara terus-menerus dan pencarian elemen dapat dilakukan secara efisien menggunakan teknik pencarian biner (binary search).