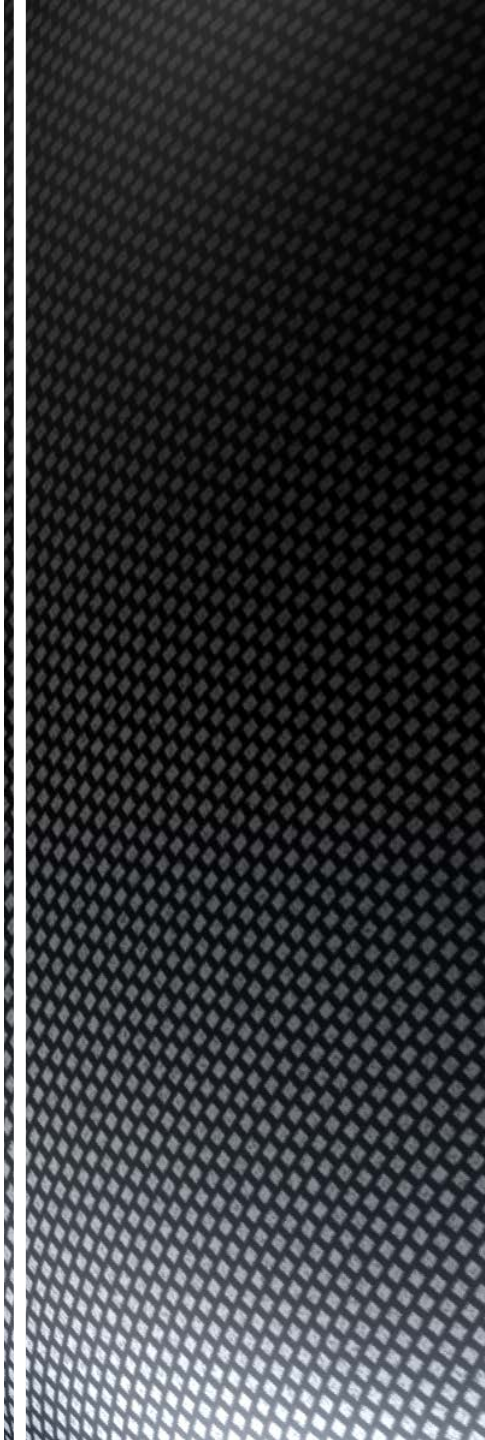
COMPSCI 107
Computer Science Fundamentals

Lecture 13 – Queues



ADT - Queue

- Ordered collection of data
  - Items are added to the back of the queue
  - Items are removed from the front of the queue
- Remove data in the same order as the data added
  - First in, first out (FIFO)
- Operations
  - Enqueue
  - Dequeue
  - Peek
  - Is_empty
  - Size

Queue Implementation

- Implementation using Python list
- What is the big-O of enqueue()?
- What is the big-O of dequeue()?

```
class QueueV1:
    def __init__(self):
        self.items = []

    def is_empty(self):
        return self.items == []

    def enqueue(self, item):
        self.items.insert(0,item)

    def dequeue(self):
        return self.items.pop()

    def size(self):
        return len(self.items)
```

Example: Printer Queue Simulation

- Problem: Can the current printer handle the task load if it were set to print with a better quality but slower page rate?

- Write a simulation which models the printing tasks as random events of various lengths and arrival times.

- The current setting on the printer is 10 pages per minute and gives lower quality printing. The output will tell us if the wait times are significantly different if we use the better quality setting on the printer (5 pages per minute).

- On average, a job is sent to the printer every 3 minutes. Each job is between 1 and 20 pages.

Printer Queue Simulation

- Run ten simulations each for 3600 (simulated) seconds. Each second there is 1 in 180 chance that a new print job of 1 to 20 pages is sent to the printer.

```
simulation(10)
 1. Average Wait:  16.90 secs.  0 tasks queued
 2. Average Wait:   2.72 secs.  0 tasks queued
 3. Average Wait:  26.75 secs.  1 tasks queued
 4. Average Wait:   3.45 secs.  0 tasks queued
 5. Average Wait:   5.60 secs.  0 tasks queued
 6. Average Wait:  22.56 secs.  0 tasks queued
 7. Average Wait:  14.71 secs.  0 tasks queued
 8. Average Wait:  25.79 secs.  0 tasks queued
 9. Average Wait:   4.28 secs.  0 tasks queued
10. Average Wait:   5.79 secs.  0 tasks queued
```

Printer Queue Simulation

- Run 10 simulations of an hour of printing

- simulate_hour function returns a tuple consisting of:
  - list containing the number of seconds that each print job waited in the queue, and
  - an integer representing the number of jobs remaining in the queue

```
def simulation(ppm):
    for n in range(1, 11):
        wait_times, jobs_queued = simulate_hour(ppm)
        average_wait = sum(wait_times) / len(wait_times)
        print('{:2}. Average Wait: {:6.2f} secs.  {} tasks queued'.format(
            n, average_wait, jobs_queued))
```

Printer Queue Simulation

```
import random
def simulate_hour(pages_per_minute):
    spooler = queue()
    pages_to_print = 0
    wait_times = []

    for current_time in range(3600):
        if random.randint(1, 180) == 1:                    #add a new job to the print queue
            pages = random.randint(1, 20)
            spooler.enqueue((pages, current_time))

        if pages_to_print <= 0 and not spooler.is_empty():     #start printing a new job
            pages_to_print, start_time = spooler.dequeue()
            wait_times += [current_time - start_time]

        pages_to_print -= pages_per_minute / 60     #pages printed each second

    return (wait_times, spooler.size())
```

Exercise

One possible implementation of the ADT Queue is given here.

Is it possible to make a more efficient implementation?

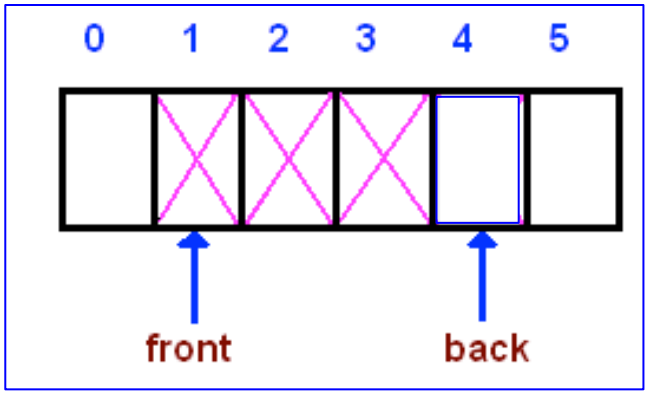
Discuss in groups how you might improve the implementation of the Queue ADT in Python.

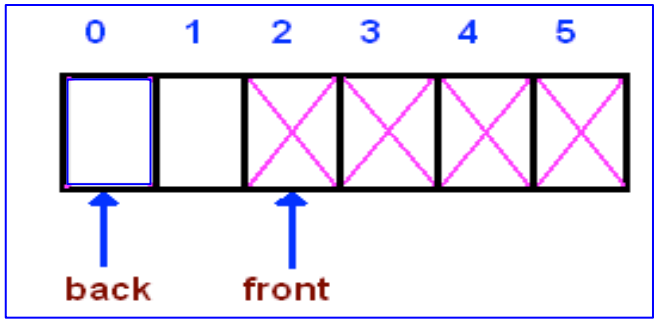
```
class QueueV1:
    def __init__(self):
        self.items = []

    def is_empty(self):
        return self.items == []

    def enqueue(self, item):
        self.items.insert(0,item)

    def dequeue(self):
        return self.items.pop()

    def size(self):
        return len(self.items)
```

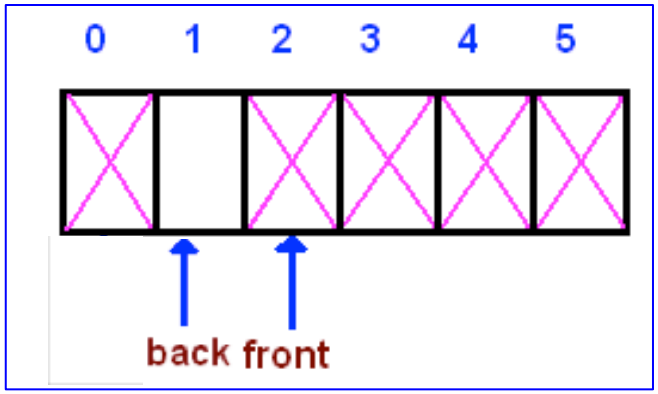
Circular Queue

- A Python list is used to store the data
  - Create an initially empty list of a given size
  - Treat the list as if it were circular
  - Keep index of the front of the queue
  - Keep index of the back of the queue

- Enqueue and Dequeue
  - Items are enqueued at the back
  - Items are dequeued at the front







Exercise

- Write the implementation for the Circular Queue. A circular queue is created by passing the constructor an initial capacity

```
q = circular_queue( 1000 )
```

```
class circular_queue:
    def __init__(self, capacity):
        #creates empty list, count, front, back

    def is_empty(self):

    def enqueue(self, item):

    def dequeue(self):

    def size():
```

Exercise

- What is the big-O running time for each of the circular_queue methods?

```
class circular_queue:
    def __init__(self, capacity):
        #creates empty list, count, front, back

    def is_empty(self):

    def enqueue(self, item):

    def dequeue(self):

    def size():
```

Exercise

- How does a user know if the circular_queue is full? What should happen when the circular_queue is full? Discuss

```
class circular_queue:
    def __init__(self, capacity):
        #creates empty list, count, front, back

    def is_empty(self):

    def enqueue(self, item):

    def dequeue(self):

    def size():
```

ADT Deque

- A Double-Ended Queue or Deque (pronounced ‘Deck’)
  - An ordered collection of items where items are added and removed from either end, either front or back

- add_front()
- add_rear()
- remove_front()
- remove_rear()
- is_empty()
- size()

Exercise

- Use a double ended queue to write a function that determines if a string is a palindrome.

- A palindrome is a sentence in which the letters appear in the same order forwards and reverse. Punctuation is ignored.

```
>>> is_palindrome('bob')
True
```

Bob – Weird Al Yankovic

I, man, am regal - a German am I
Never odd or even
If I had a hi-fi
Madam, I'm Adam
Too hot to hoot
No lemons, no melon
Too bad I hid a boot
Lisa Bonet ate no basil
Warsaw was raw
Was it a car or a cat I saw?

Rise to vote, sir
Do geese see god?
"Do nine men interpret?" "Nine men," I nod
Rats live on no evil star
Won't lovers revolt now?
Race fast, safe car
Pa's a sap
Ma is as selfless as I am
May a moody baby doom a yam?

Ah, Satan sees Natasha
No devil lived on
Lonely Tylenol
Not a banana baton
No "x" in "Nixon"
O, stone, be not so
O Geronimo, no minor ego
"Naomi," I moan
"A Toyota's a Toyota"
A dog, a panic in a pagoda

Oh no! Don Ho!
Nurse, I spy gypsies - run!
Senile felines
Now I see bees I won
UFO tofu
We panic in a pew
Oozy rat in a sanitary zoo
God! A red nugget! A fat egg under a dog!
Go hang a salami, I'm a lasagna hog