ChoMan Cho 20200225

Table of Contents

ChoMan Cho 20200225

Introduction

In this assignment, we are going to implement the linked lists, stack and queue data structure using Python. This report is going to implement data structure of linked list, stack and queue using Python. Then it will discuss stack ADT and queue ADT. List the key operations and support operations associated with these ADTs. Examples of real world will also be given. For part two, it adopts ADT concepts of Part 1 to provide a complete implementation of a stack ADT and to examine the size of stack. For part three, it adopts ADT concepts of Part 1 to provide a complete implementation of a queue ADT and to examine the size of queue by looking at the example. For part two and three, please look at the jupyter notebook for further explanation.

Linked List

A linked list is an abstract data type which links nodes sequentially using pointers. Each node contains a value and a pointer. A node is connected with each other with pointer. The structure of linked list facilitates easy insertion or removal of nodes from lists as it traverses from head of linked list. The attribute of linked list is that it is not stored in a contiguous block of memory, unlike array. Therefore, resizing of the list is faster than array as array is contiguous block of memory and it will eventually get filled or have to be resized (a costly operation that isn't always possible). In the example, linked list will used to illustrate the example of queue and stack as it is a faster way compared to array for its nature.

Stack ADT

Stack ADT is an abstract data type that elements are placed on top of each other. And all new elements are added to the top of stack. A stack is a groups of objects that inserted and removed at the top of stack. This follows the last in first out (LIFO) principle. Therefore, a stack is ordered in sequence based on how long they have been in stack. Any key operations are going to start from the top of elements, for example, a push operation is going to insert element at the top of stack. A stack ADT can be implemented using arrays or linked lists.

Operations

Key operations performed by stack ADT:
push() – Insert an element at the top of stack
pop() – remove an element at the top of stack

Support operations performed by stack ADT:
top() – peek at the top of element and return the result
size() – return the number of elements in a stack
isEmpty() – return true if the stack is empty, else return false

ChoMan Cho 20200225

Stack ADT Examples

1. A firearm magazine is a best example of stack.
It is easy to see that bullets can be inserted from the top, and only the top bullet is used every time a gun fire. A magazine works with LIFO (Last In First Out).

2. View history stored in web browsers. Web browsers worked with LIFO to keep the addresses of the sites that the user visited. Every time when a user visit a website, the history of that website is stored by 'pushing' to the stack. When user clicked back button, key operation of pop is used to retrieve the history.

Queue ADT

A queue is an abstract type that insertion takes place only at the tail, and removal takes place only at the head. In a queue, the insert operation is implemented at a position that known as 'rear'(enqueued) and the delete operation is implemented at a position that known as 'front'(dequeued). It follows example of first in first out(FIFO). A queue ADT can be implemented using arrays or linked lists.

Operations

Key operation can be performed by stack ADT:
enqueue() – Insert an element at the end of the queue
dequeue() – Remove first element of the queue, if the queue is not empty.

Support operations performed by stack ADT:
is_Empty() - return true if the stack is empty, else return false
size() - return the number of elements in a queue
top() - return the first element of queue if there is no empty list

Queue ADT Examples

1. Print spooling allows users to send a multiple documents to a printer with a queue ADT applied. In print spooling, documents loaded into a buffer and printer takes it from the first elements to the second element one by one in a queue. It follows FIFO principle.

2. During the pandemic, we see a queue at the front door of supermarket to facilitate the social distancing. The queue follows FIFO principle which first one at the queue go into the shop first and second and third till the last one of element.