Midterm Review -2

Yan Yan

Contents

1. Stacks
2. Queues
3. Searching

Stack: Definition

- **Stack**: A stack is an ADT in which items are only inserted on or removed from the top of a stack.
- **Push** operation inserts an item on the top of the stack.
- **Pop** operation removes and returns the item at the top of the stack.
- A *linear* data structure, in which elements are accessed using the LIFO (Last in First Out) Order.

Push

- Push: Adds an item to the stack. If the stack is full, then it is said to be an overflow condition.

Algorithm for push:

```
begin
 if stack is full
    return
 endif
else
   increment top
   stack[top] assign value
end else
end procedure
```

Pop

- Pop: Removes an item from the stack. The items are popped in the reversed order in which they are pushed. If the stack is empty, then it is said to be an **underflow** condition.

Algorithm for pop:

```
begin
 if stack is empty
    return
 endif
else
   store value of stack[top]
   decrement top
   return value
end else
end procedure
```

Other Operations

- **isEmpty**: Returns true if the stack is empty, else false.
- **isFull**: Returns true if the stack is full, else false.
- **Peek** or **Top**: Returns the top element of the stack.

Time Complexities

- What are the time complexities of the operations on the stack?

- push(), pop(), isEmpty() and peek() all take O(1) time. We do not run any loop in any of these operations.

Stack Linked List Implementation

- **Head** of the list is the top of the stack.
- Push/Pop: update the top of the stack. Similar to adding/deleting the first node in a linked list.

Queue: Definition

- **Queue**: A queue is an ADT in which items are inserted at the end of the queue and removed from the front of the queue.
- **Enqueue** operation inserts an item at the end of the queue.
- **Dequeue** operation removes and returns the item at the front of the queue.
- A *linear* data structure, FIFO (First in First Out).

Time Complexities

- What are the time complexities of the operations on the queue?
- We require Enqueue(), Dequeue(), isEmpty() and peek() all take O(1) time.

Linked-List Implementation

- LL's head node – queue's front
- LL's tail node – queue's rear
- Enqueue – append to the end of LL
- Dequeue – remove the head node

- Refer to these operations in the linked list.

Sample Short Answer Questions

- The operation used to **add** an item to the **stack** is called push
- The operation used to **remove** an item to the **stack** is called pop
- The operation used to **add** an item to the **queue** is called en
- The operation used to **remove** an item to the **queue** is called de

Sample Long Answer Questions

- Explain the differences between a stack and a queue regarding how they handle data in terms of removal and addition.

Sample Long Answer Questions

- Given numQueue: 1, 4, 7, 8 with 1 at the front, and the following operations:

Enqueue(numQueue, 2)
Dequeue(numQueue)

- What is the numQueue after the operations?
- If this queue is implemented by linked list, what's the head pointing to after the operations?

Linear Search

- Linear search is a search algorithm that starts from the beginning of a list, and checks each element until the search key is found or the end of the list is reached
  - mostly used to search an unordered list of elements

Binary Search

- If the list is **sorted** and directly accessible (such as an **array**), we could use the binary search to speed up
  - Search starts with the middle element
  - If the search key is found, the algorithm returns the matching location.
  - If the search key is not found, the algorithm repeats the search on the remaining left sublist or the remaining right sublist

Binary Search

- If initial length of array =n
  - Iteration 1 - Length of array=n/2
  - Iteration 2 - Length of array =(n/2)/2=n/ 2^2
  - …
  - Iteration k - Length of array =n/2^k
  - After k iterations, the size of the array becomes 1
  - Time Complexity O($\log_2 n$).

Interpolation Search

- Improved version of binary search for uniformly distributed elements
  - "Guess" the position of the searched elements
  - The calculation is done based on the values at the bounds of the search space and the value to be searched.
    - Usually called a prob
- If prob is the searched element, return; or narrow down the search space base on the prob

Interpolation Search

- Example A=[1,3,5,7,9,11], x=3
- $prob = low + \frac{(x - A[low])(high - low)}{(A[high] - A[low])}$
- $prob = 0 + \frac{(3-1)(5-0)}{(11-1)} = 1$

Recursive Algorithm

- A recursive algorithm is an algorithm that breaks the problem into *smaller subproblems* and applies the algorithm itself to solve the smaller subproblems.

Recursive Algorithm

- Example

```
int sum(int n) {
        if (n != 0)
        // sum() function calls itself
                return n + sum(n-1);
        else
                return n;
}
```

Sample Short Answer Questions

- What is the function return when n=4?

```
int func (int n) {
        if (n==0) {
                return 0;
        }
        else if (n == 1) {
                return 1;
        }
        else {
                return func(n-1)+ func(n-2);
        }
}
```

Sample Short Answer Questions

- Name one search or sort algorithm that can be implemented by recursion

Sample Long Answer Questions

- A list of 32 students is stored in an array (by unique student number), how many times would a binary search algorithm need to split the array in half in order to find a certain student? Discuss the best and the worst case separately including calculation steps.

More on Midterm Questions

- Please review ALL questions/examples/exercise in the lecture slides, labs, and A1.