Lecture 13

Thursday Oct. 19

Example: Determining Asymptotic Running Time

```
1  containsDuplicate (int[] a, int n) {
2    for (int i = 0; i < n; ) {
3      for (int j = 0; j < n; ) {
4        if (i != j && a[i] == a[j]) {
5          return true; }
6        j ++; }
7      i ++; }
8    return false; }
```

O(1)

$\bar{i} = n-1$

$\bar{j} = 0 \ldots n-1$ (n)

Body of loop takes O(1) $\begin{cases} \bar{i} = 0 \\ \bar{j} = 0 \ldots n-1 \end{cases}$ (n)

How many times? n^2 $\begin{cases} \bar{i} = 1 \\ \bar{j} = 0 \ldots n-1 \end{cases}$ (n)

RT: $O(1) * n^2 = O(n^2)$

Example: Determining Asymptotic Running Time

```
1  sumMaxAndCrossProducts (int[] a, int n) {
2    int max = a[0];                                  O(1)
3    for(int i = 1; i < n;) {                         O(n)
4      if (a[i] > max) { max = a[i]; }
5    }
6    int sum = max;                                   O(1)
7    for (int j = 0; j < n; j ++) {                   O(n^2)
8      for (int k = 0; k < n; k ++) {
9        sum += a[j] * a[k]; } }
10   return sum; }                                    O(1)
```

RT: $O(n + 1 + n^2 + 1) = O(n^2)$.

n^2 dominates the RT

Example: Determining Asymptotic Running Time

```
1  triangularSum (int[] a, int n) {
2    int sum = 0;
3    for (int i = 0; i < n; i ++) {
4      for (int j = i; j < n; j ++) {
5        sum += a[j]; } }
6    return sum; }
```

$\bar{I} = n-1$

$\bar{J} = n-1 \ldots n-1$ (1)

$O(1)$

$\bar{J} = \bar{I}$

$\bar{I} = 0$

$\bar{J} = \frac{\bar{I}}{0} \ldots n-1$ (n)

How many iterations?

$n + (n-1) + (n-2) + \ldots + 1 = \frac{n \cdot (n+1)}{2}$

$O(n^2) = n^2$

$\bar{I} = 1$

$\bar{J} = 1 \ldots (n-1)$ (n-1)

Running Time of Inserting into an Array [a, b]

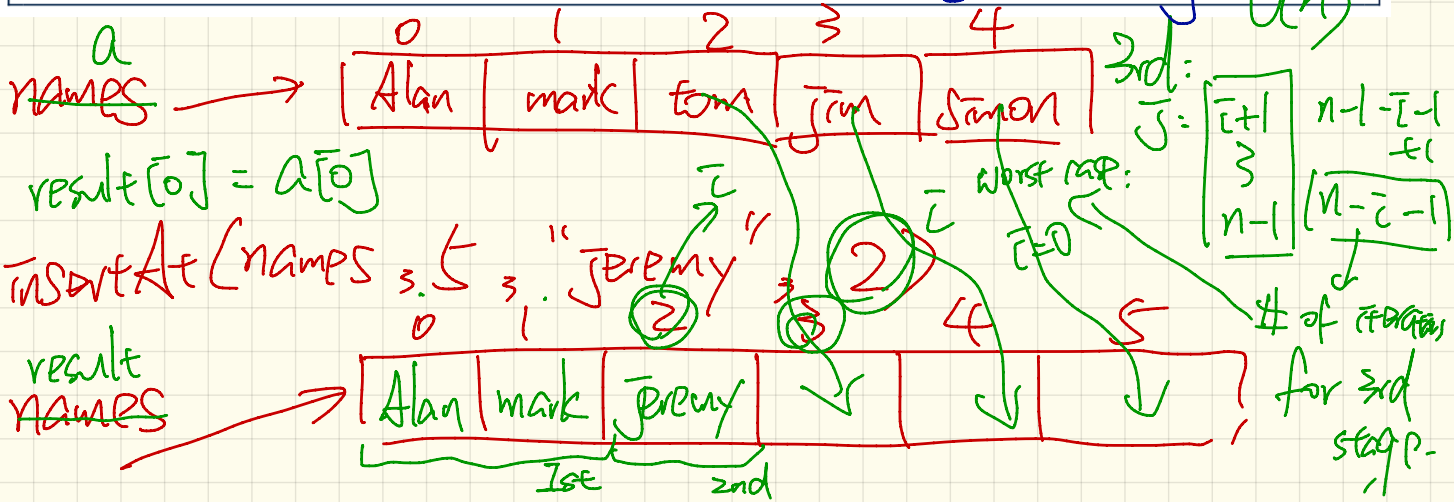
```
insertAt(String[] a, int n, String e, int i)
  String[] result = new String[n + 1];
  for(int j = 0; j < i; j ++){ result[j] = a[j]; }
  result[i] = e;
  for(int j = i + 1; j < n; j ++){ result[j] = a[j-1]; }
  return result;
```

a.length

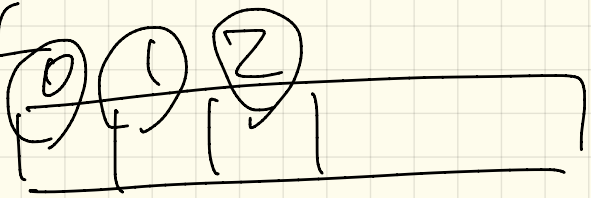
1st: O(1)

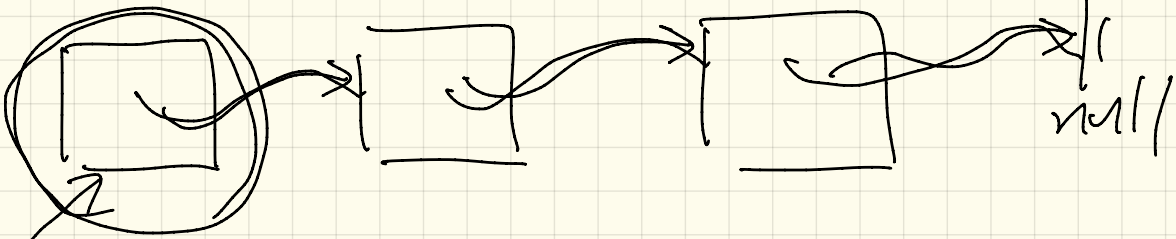
2nd: O(1) — depends on i

3rd: O(n)



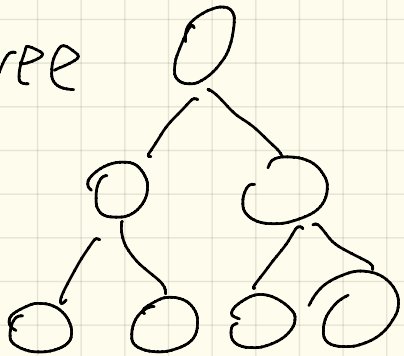
linear



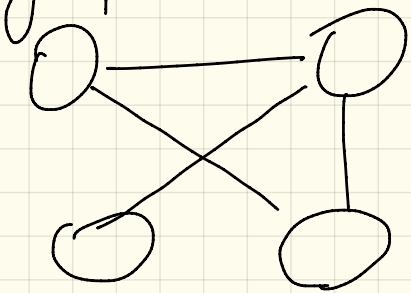


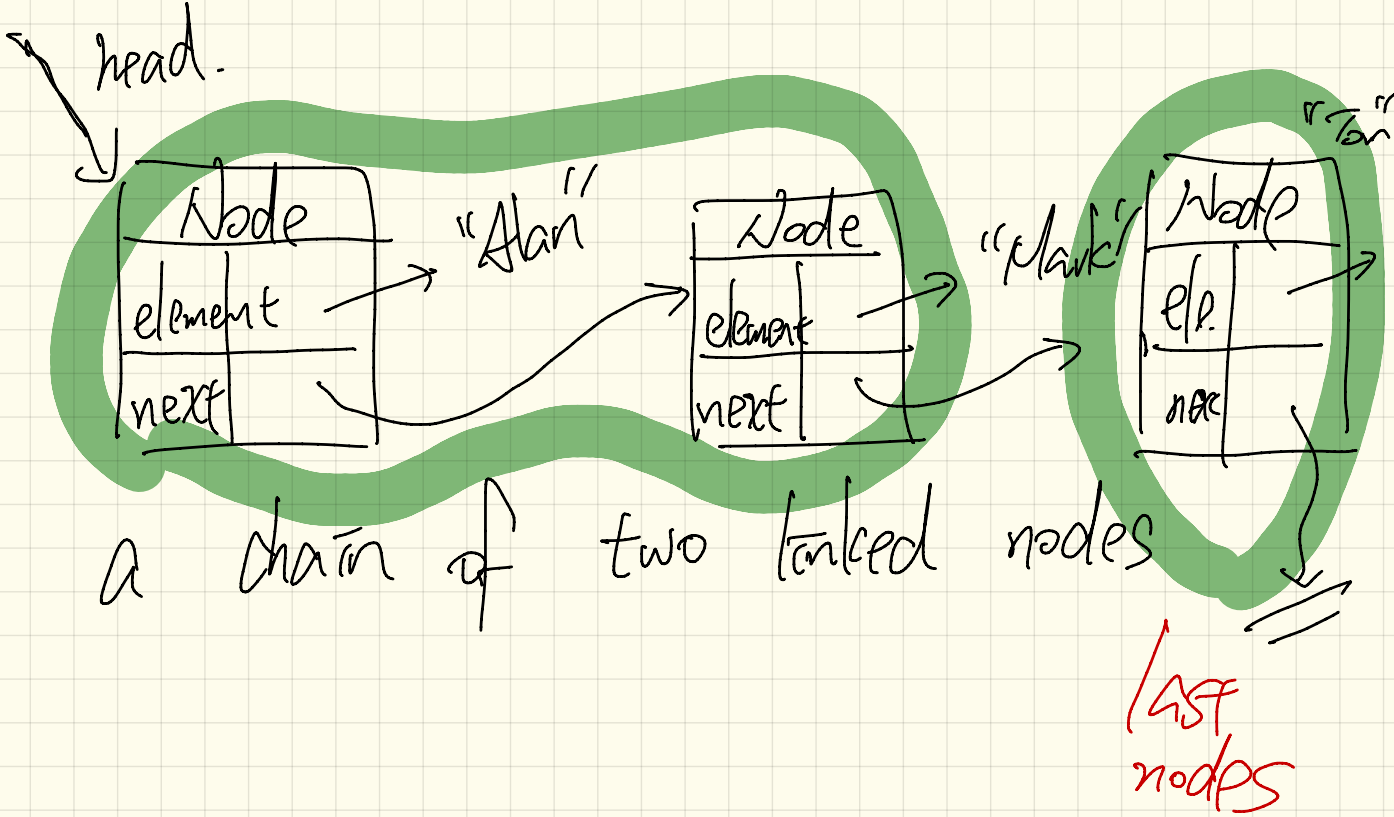
non-linear

tree

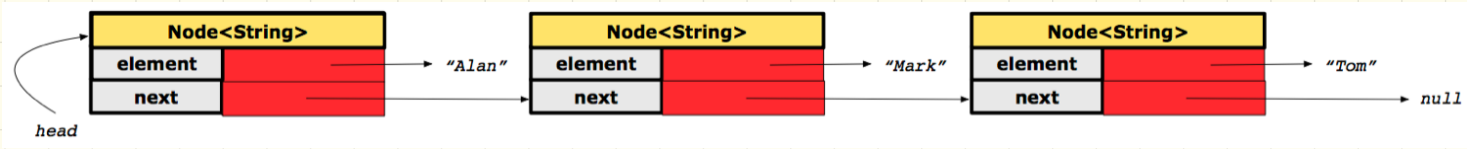


graph





a chain of two linked nodes




```
public class Node {
   private String element;
   private Node next;
   public Node(String e, Node n) { element = e; next = n; }
   public String getElement() { return element; }
   public Node getNext() { return next; }
   public void setNext(Node n) { next = n; }
}
```

mark

alan.next == mark

Node alan = new Node ("Alan", mark)

Approach 1

```
Node tom = new Node("Tom", null);
Node mark = new Node("Mark", tom);
Node alan = new Node("Alan", mark);
```