w14-Lec

Sorting and Searching

Assembled for 204111

by Kittipitch Kuptavanich

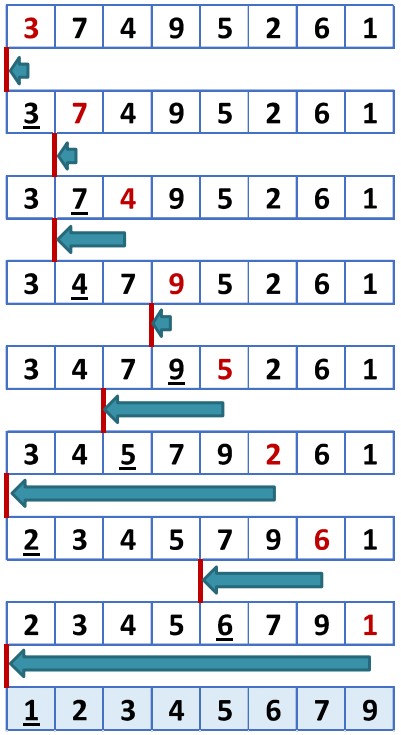
Sorting

- **Sorting คือเรียงสมาชิกของรายการ (list) ให้เป็นไปตามรูปแบบของอันดับที่กำหนด**
  - **ส่วนใหญ่อันดับที่ใช้กันคือ**
    - **อันดับตัวเลข**
    - **และอันดับตัวอักษร**
- **สิ่งที่ต้องคำนึงถึง**
  - **ความถูกต้อง (this class)**
  - ความซับซ้อนในการคำนวณ (ระยะเวลา)
  - การใช้หน่วยความจำ
  - ...

Efficiency

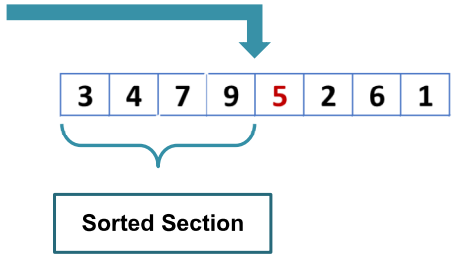
Efficiency

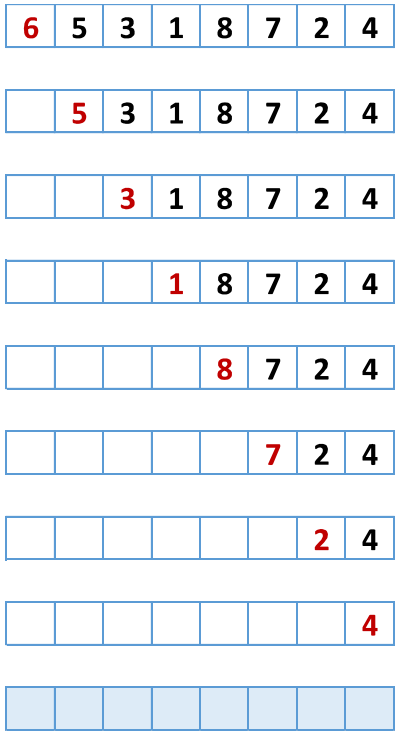
- **ประสิทธิภาพของโปรแกรมพิจารณาจากการใช้ทรัพยากรต่าง ๆ ของโปรแกรม (เมื่อ input มีขนาดใหญ่) เช่น**
  - **เวลา**
  - **พื้นที่ (memory)**
  - **Bandwidth..**
  - **อื่น ๆ**



Insertion Sort

- **เริ่มจากตำแหน่งซ้ายสุด**
- **จับวิ่งไปทางซ้ายจนพบค่าน้อยกว่า**





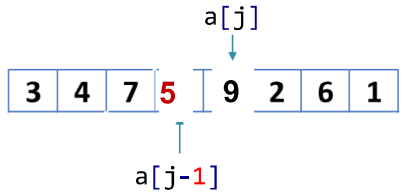
Practice 1: Insertion Sort

Insertion Sort [2]

```
a = [6,5,3,1,8,7,2,4]
size = length(a)


for i in 1 … size
    j = i
    while j > 0 and a[j] < a[j-1]
        swap(a[j],a[j-1])
        j-=1


print a
```



Insertion Sort [4]

Advantages:

- ง่ายต่อการ implement
  Simple implementation
- มีประสิทธิภาพดีกับข้อมูลขนาดเล็ก
  Efficient for (quite) small data sets
- ใช้ได้ดีแม้ data มีการเรียงลำดับมาแล้วบางส่วน
  Adaptive, i.e., efficient for data sets that are already substantially sorted:

Insertion Sort [5]

Advantages (cont'd):

- ไม่เปลี่ยนลำดับของสิ่งที่ต้องการเรียงหาก key มีค่าเท่ากัน
  Stable; i.e., does not change the relative order of elements with equal keys
- ใช้พื้นที่เพิ่มเติมในการ sort ที่คงที่ (ไม่ขึ้นกับขนาดข้อมูล)
  In-place; i.e., only requires a constant amount of additional memory space
- สามารถเรียงข้อมูลไปพร้อม ๆ กับรับข้อมูล (ไม่ต้องรับเข้ามาหมดก่อนถึงจะดำเนินการได้)
  Online; i.e., can sort a list as it receives it

Merge Algorithm

- ในกรณีที่มี list ที่มีการเรียงลำดับไว้แล้วมากกว่าหนึ่ง list
- การนำ list ทั้งสองมารวมกันเพื่อให้ได้ list ใหม่ที่มีการเรียงลำดับ
  - ซึ่งประกอบด้วย element ทั้งหมดจาก list ทั้งสอง
  - เรียกว่า merging
- โดยวิธีในการรวม list ลักษณะดังกล่าว เรียกว่า Merge Algorithm

Merge Algorithm

1	3	4	5	6

2	5	7	9	10

- ให้ index i และ j ชี้ที่ตำแหน่ง head ของ list A และ list B
- ให้ C เป็น list ว่าง
- ให้ a และ b เป็น value ใน list A และ B ที่ตำแหน่ง i และ j ตามลำดับ
- If $a < b$ เพิ่ม a ไปที่ list C และ increment i
  - Else เพิ่ม b ไปที่ list C และ increment j
- ถ้า list ใดหมดก่อน ให้นำ element ทั้งหมดของ list ที่เหลือ เพิ่มไปที่ list C

Merge Algorithm

a

1	3	4	5	6

2	5	7	9	10

1

b

1	3	4	5	6

2	5	7	9	10

1	2

c

1	3	4	5	6

2	5	7	9	10

1	2	3

d

1	3	4	5	6

2	5	7	9	10

1	2	3	4

e

1	3	4	5	6

2	5	7	9	10

1	2	3	4	5

f

1	3	4	5	6

2	5	7	9	10

1	2	3	4	5	5

g

1	3	4	5	6

2	5	7	9	10

1	2	3	4	5	5	6

h

1	3	4	5	6

2	5	7	9	10

1	2	3	4	5	5	6	7	9	10

Merge Algorithm

```
Function mergeList(A,B)
    m = len(A)
    n = len(B)
    i = 0
    j = 0
    C = []
    while i < m and j < n
        if A[i] < B[j]
            C.append(A[i])
            i+=1
        else
            C.append(B[j])
            j+=1

    if i < m
        C.extend(the_rest(A,i))
    if j < n
        C.extend(the_rest(B,j))

    return C

A = [1,3,4,5,6]
B = [2,5,7,9,10]

print(mergeList(A,B))
```

SEARCHING

Linear Search

```
Function linearSearch(array A, key)
    size = length(A)
    for i in 0 … size
        if key == A[i]
            return i
    return -1


A = [1,14,3,18,2,6,5]

print(linearSearch(A,6))
```


Reference

- http://www.kosbie.net/cmu/fall-12/15-112/handouts/notes-efficiency.html
- http://en.wikipedia.org/wiki/Insertion_sort
- http://en.wikipedia.org/wiki/Merge_algorithm