- 作业讲解
  - CS第5.6节问题4、8
  - CS第5.7节问题1、2、4、6、12、16、18
  - TC第5.2节练习1、2、4
  - TC第5.3节练习1、2、3、4
  - TC第5章问题2

CS第5.6节问题4

- $E(M(t)) = \sum_{t=1}^{\infty} P(t) \cdot E(M(t) \mid t)$

$$= \sum_{t=1}^{\infty} \left(\frac{1}{4}\right)^{t-1} \frac{3}{4} \cdot \left(1(t-1) + \left(2\frac{1}{3} + 3\frac{1}{3} + 4\frac{1}{3}\right)\right)$$

$$= \sum_{t=1}^{\infty} \left(\frac{1}{4}\right)^{t-1} \frac{3}{4} \cdot (t-1+3)$$

$$= \frac{3}{4} \sum_{t=1}^{\infty} (t-1) \left(\frac{1}{4}\right)^{t-1} + \frac{9}{4} \sum_{t=1}^{\infty} \left(\frac{1}{4}\right)^{t-1}$$

$$= \frac{3}{4} \sum_{t=0}^{\infty} (t) \left(\frac{1}{4}\right)^{t} + \frac{9}{4} \frac{1}{1-\frac{1}{4}}$$

$$= \frac{3}{4} \frac{\frac{1}{4}}{\left(1-\frac{1}{4}\right)^2} + 3$$

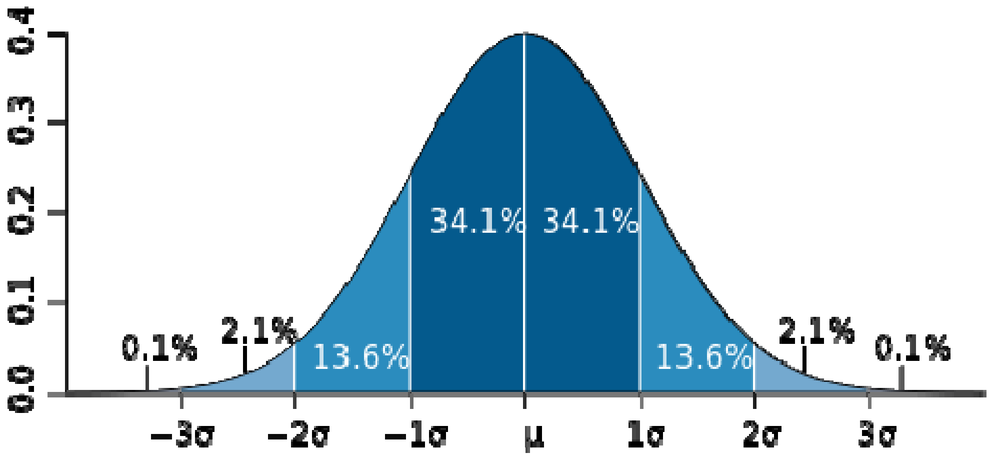
$$= \frac{10}{3}$$

CS第5.6节问题8

- $\sum_{i=1}^{n} E(X \mid F_i)P(F_i) = \sum_{i=1}^{n}\sum_{s\in F_i} X(s)\frac{P(s)}{P(F_i)}P(F_i) = \sum_{i=1}^{n}\sum_{s\in F_i} X(s)P(s) = \sum_{s\in S} X(s)P(s) = E(X)$

CS第5.7节问题12

- 95%：$2\sigma = 2\cdot 0.4\sqrt{n} = 5\%\cdot n \Rightarrow n = 256$



CS第5.7节问题16a

- $$V(X) = \sum_{i=1}^{n} P(x_i)(X(x_i) - E(X))^2$$

  $$\geq \sum_{i=1}^{k} P(x_i)(X(x_i) - E(X))^2$$

  $$> \sum_{i=1}^{k} P(x_i) r^2$$

  $$= r^2 \sum_{i=1}^{k} P(x_i)$$

  $$= r^2 P(E)$$

TC第5.2节练习1

- Probability that you hire exactly one time
  - 1/n
- Probability that you hire exactly n times
  - 1/n!

TC第5.2节练习2

- Exactly twice
  - 第一个不是最优
  - 最优之前没有比第一个更优的

$$\sum_{i=1}^{n-1} P\left(\text{第一个是}x_i\right) P\left(x_{i+1} \ldots x_{n-1}\text{都在}x_n\text{之后} \mid \text{第一个是}x_i\right)$$

$$= \sum_{i=1}^{n-1} \frac{1}{n} \frac{1}{n-i}$$

$$= \frac{1}{n} \sum_{i=1}^{n-1} \frac{1}{n-i}$$

$$= \frac{1}{n} \sum_{i=1}^{n-1} \frac{1}{i}$$

TC第5.2节练习4

- Expected number of customers who get back their own hat
  - Indicator random variable X_i
  - $E(X)=E(\sum X_i)=\sum E(X_i)=\sum P(X_i=1)=\sum(1/n)=1$

TC第5.3节练习1

```
RANDOMIZE-IN-PLACE(A)
1  n = A.length
2  for i = 2 to n                        ← swap A[1] with A[RANDOM(1, n)]
3      swap A[i] with A[RANDOM(i, n)]
```

Just prior to the ith iteration of the **for** loop of lines 2–3, for each possible $(i-1)$-permutation of the n elements, the subarray $A[1..i-1]$ contains this $(i-1)$-permutation with probability $(n-i+1)!/n!$.   i=1?

Professor Marceau objects to the loop invariant used in the proof of Lemma 5.5. He questions whether it is true prior to the first iteration. He reasons that we could just as easily declare that an empty subarray contains no 0-permutations. Therefore, the probability that an empty subarray contains a 0-permutation should be 0, thus invalidating the loop invariant prior to the first iteration. Rewrite the procedure RANDOMIZE-IN-PLACE so that its associated loop invariant applies to a nonempty subarray prior to the first iteration, and modify the proof of Lemma 5.5 for your procedure.

TC第5.3节练习2

Professor Kelp decides to write a procedure that produces at random any permutation besides the identity permutation. He proposes the following procedure:

```
PERMUTE-WITHOUT-IDENTITY(A)
n = A.length
for i = 1 to n - 1
    swap A[i] with A[RANDOM(i + 1, n)]
```

Does this code do what Professor Kelp intends?

- A[1]总是被换掉

TC第5.3节练习3

```
PERMUTE-WITH-ALL(A)
n = A.length
for i = 1 to n
    swap A[i] with A[RANDOM(1, n)]
```

Does this code produce a uniform random permutation? Why or why not?

- 方法一
  - n^n种输出（可能有重复）
  - n!种permutation
  - 无法整除
- 方法二
  - n=3时，identity permutation出现的概率是4/27，而非1/6

TC第5.3节练习4

```
PERMUTE-BY-CYCLIC(A)
n = A.length
let B[1..n] be a new array
offset = RANDOM(1, n)
for i = 1 to n
    dest = i + offset
    if dest > n
        dest = dest - n
    B[dest] = A[i]
return B
```

Show that each element $A[i]$ has a $1/n$ probability of winding up in any particular position in B. Then show that Professor Armstrong is mistaken by showing that the resulting permutation is not uniformly random.

- 只有n种输出

TC第5章问题2

- (d) expected number of indices … have checked all elements
  - X_i：从有i-1个以后到第i个出现的次数
  - $E(X_i)=n/(n-(i-1))$
  - $E(X)=E(\sum X_i)=\sum E(X_i)=n\sum 1/(n-i+1)=n\sum(1/i)$
- (f) k≥1 indices … average-case running time of DETERMINISTIC-SEARCH
  - $$\sum_{i=1}^{n-(k-1)} i \cdot \frac{\binom{n-i}{k-1}}{\binom{n}{k}}$$

- 教材讨论
  - TC第10章

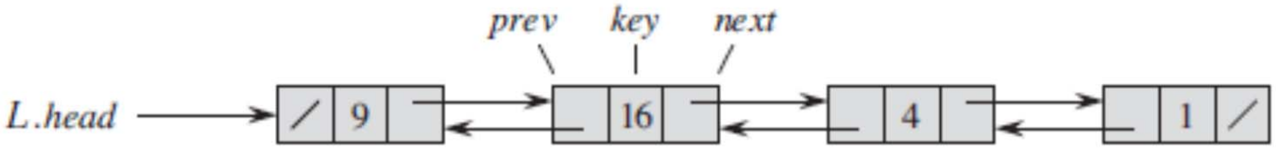
问题1：dynamic set及其实现

- 什么是dynamic set，什么又是dictionary？
- 你理解dynamic set的这些操作了吗？
  其中，哪些是query，哪些是modification？

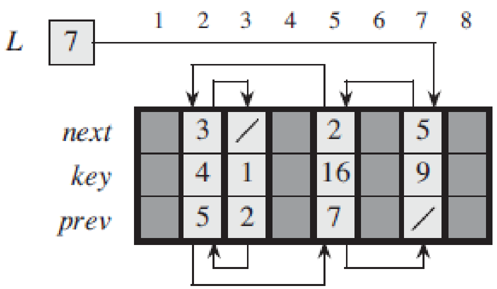
Search
Insert
Delete
Minimum
Maximum
Successor
Predecessor

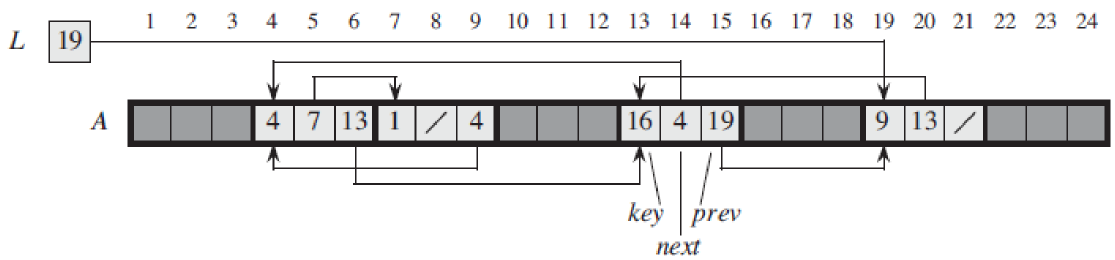
问题2：linked list

- 你怎么理解singly/doubly/circular linked list？



- 你能正确写出doubly linked list的insert/delete操作吗？
- 你能正确写出singly linked list的insert/delete操作吗？
- 你能比较linked list的两种数组实现的优劣吗？

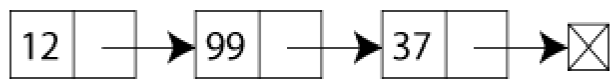




问题2： linked list (续)

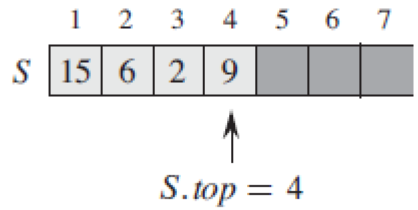
- 你能利用linked list实现dynamic set的所有操作吗？
  它们的运行时间分别是多少？

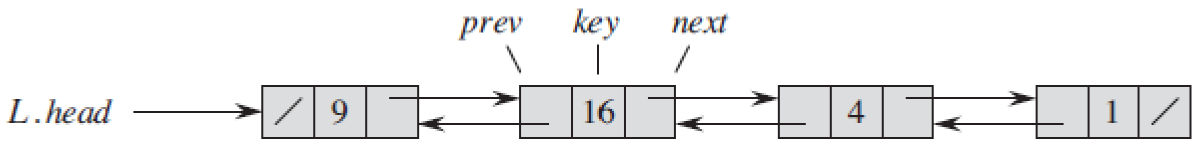
Search
Insert
Delete
Minimum
Maximum
Successor
Predecessor



问题3： stack

- 你怎么理解stack？
- 你能正确写出stack的数组实现的push/pop操作吗？
- 你能用linked list来实现stack吗？

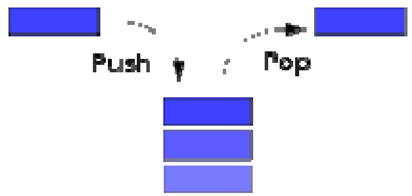




问题3： stack (续)

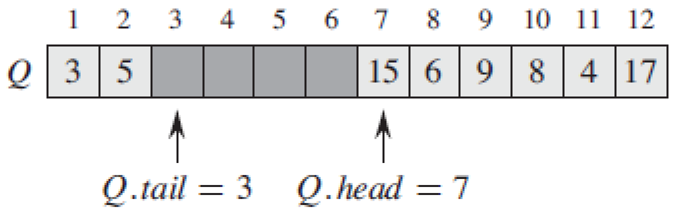
- 你能利用stack实现dynamic set的所有操作吗？它们的运行时间分别是多少？

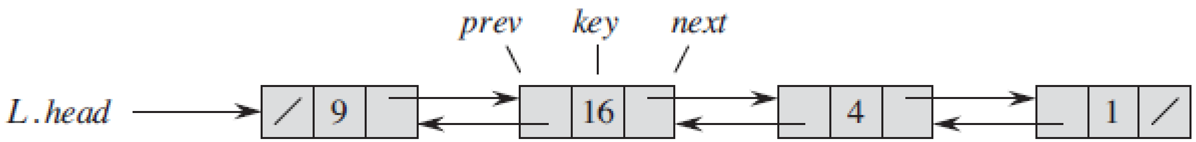
Search
Insert
Delete
Minimum
Maximum
Successor
Predecessor



问题4：queue

- 你怎么理解queue？
- 你能正确写出queue的数组实现的enqueue/dequeue操作吗？
- 你能用linked list来实现queue吗？

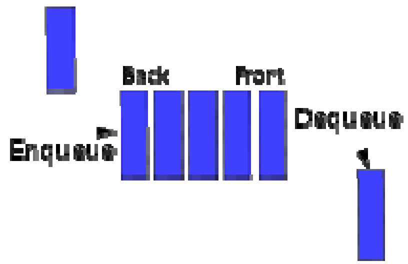




问题4：queue (续)

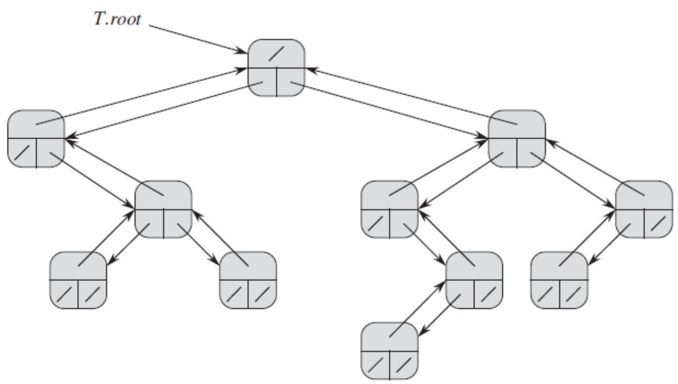
- 你能利用queue实现dynamic set的所有操作吗？它们的运行时间分别是多少？

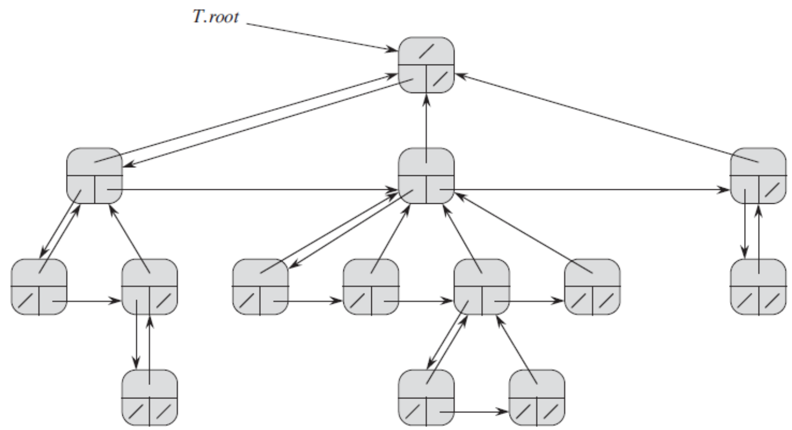
Search
Insert
Delete
Minimum
Maximum
Successor
Predecessor



问题5： rooted tree

- 你怎么理解rooted tree？
- binary tree和linked list有什么异同？
- 你怎么理解left-child, right-sibling representation？
- 你能用数组来实现left-child, right-sibling representation吗？

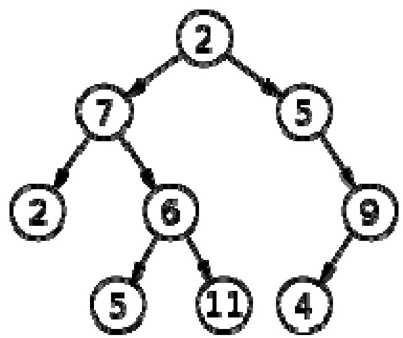




问题5： rooted tree (续)

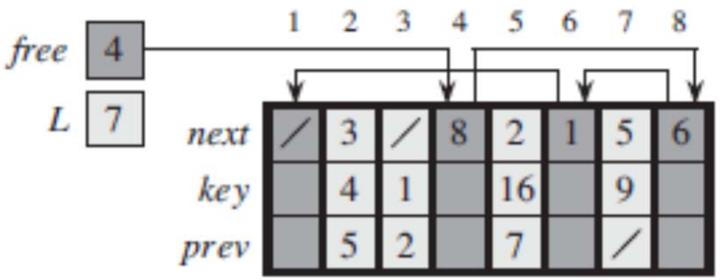
- 你能利用binary tree实现dynamic set的所有操作吗？它们的运行时间分别是多少？

Search
Insert
Delete
Minimum
Maximum
Successor
Predecessor



问题6：allocating and freeing objects

- 为什么需要allocating and freeing objects？
- free list本质上是一种什么样的数据结构？



- 你怎么理解service several linked lists with a single linked list？
- 你觉得这种做法有什么优缺点？