2. Data Structures

Fred

- Elementary Data structure
- Object Libraries
- Program Design Example: Going to War
- Hitting the Deck
- String Input/Output
- Winning the War
- Testing and Debugging

2. Data Structure

- Selecting the right data structure can make big differences in the complexity of the resulting implementation!
- In This Chapter, we will review the fundamental data structures.

2.1 Elementary Data Structures

- Stack
  - Last in - first out
  - Operations:
    - Push(x, s)
    - Pop(s)
    - Initialize(s)
    - Full(s), Empty(s)
  - Example: dinner plates

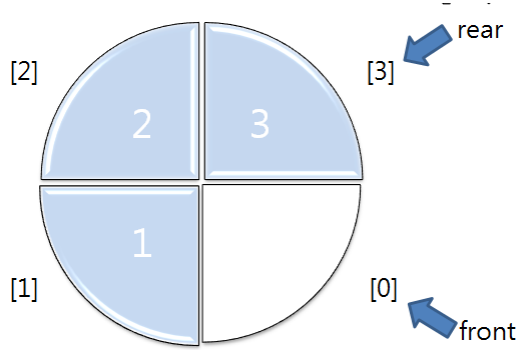
  - 다음에 사용할 plate가 어떤 것이든 상관 없을 때 사용.
  - Array나 linked list로 구현하면 된다.
    - Linked list로 구성하면 overflow가 일어나지 않는다!

Array List

Linked List

2.1 Elementary Data Structures

- Queue
  - Frist in - first out
  - Breadth-first searc에 사용된다.
  - Operations:
    - Enqueue(x, q)
    - Dequeue(q)
    - Initialize, full, empty
  - 구현 방법
    - Array
      - Dequeue를 실행할 때 모든 element들을 평행이동해야한다. -> 비효율적.
      - We can maintain indices to the first (head) and last (tail) elements in the array/queue and do all operations locally.
      - Circular queue: the pointer to the front of the list is always behind the back pointer!



Enqueue 3

Circular queue

2.1 Elementary Data Structures

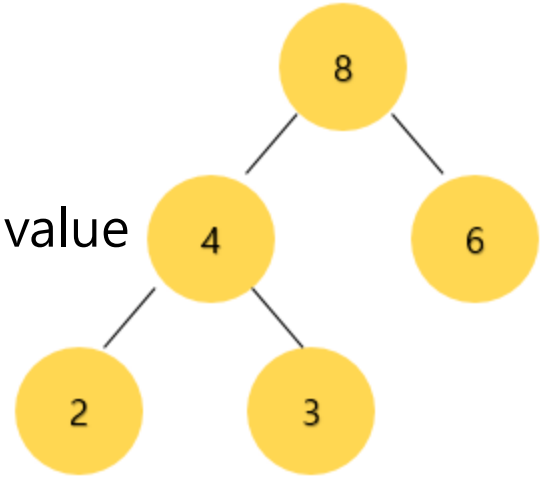
- Dictionaries
  - Content-based 입력 / 출력
  - Operations
    - Insert(x, d)
    - Delete(x, d)
    - Search(k, d), key k
  - Static Dictionaries
    - 한 번 구성되면 바뀌지 않음. Search만 가능.
    - Array로 쉽게 만들 수 있다. Sort 여부를 선택 가능.

2.1 Elementary Data Structures

- Dictionaries
  - Semi-dynamic Dictionaries
    - Insertion and Search quearies가 가능하지만 deletion 불가
    - Hash table의 structure에 Semi-dynamic Dictionary를 적용한다고 해보자.
    - Hashing의 두 요소는 아래와 같다.
      - Key와 hash를 잇는 hashing function을 정의한다.
      - Hashing function의 value가 index와 매칭되도록 array를 만든다.
    - Steve를 alphabet-size number system으로 바꾸면 아래와 같다.
      - "steve" ⇒ 26^4 *x*19+26^3 *x*20+26^2 *x*5+26^1 *x*22+26^0 *x*5 (26진수!)
    - Deletion은 insertion chain을 깨뜨리므로 ictionary의 처리 로직을 복잡하게 만든다.
  - Fully-dynamic Dictionaries
    - Deletion까지 포함하는 Dictionary

2.1 Elementary Data Structures

- Priority Queues
  - 다음 세 operation을 지원하는 queue를 priority Queue 라고 한다.
    - Insert(x, p): p에 x를 넣는다.
    - Maximum(p): return the item with largest key
    - ExtractMax(p): return&remove the item with the largest key
  - Priority queues are used to maintain schedules and calendars.
    - 이땐 priority가 시간 순이겠죠?
  - Example: binary heap
    - Tree-like data structure to find max or min value



2.1 Elementary Data Structures

- Sets
  - unordered collections of elements drawn from a given universal set U
  - Operations
    - Member(x, S): bool, is an item x an element of subset S?
    - Union(A, B): Construct $A \cup B$
    - Intersection(A, B): Construct $A \cap B$
    - Insert(x, S), Delete(x, S) – insert, delete element x

2.2 Object Libraries

- C++
  - Stack: S.push(), S.top(), S.pop(), S.empty()
  - Queue: Q.front(), Q.back(), Q.push(), Q.pop()
  - Dictionaries: H.erase(), H.find(), H.insert()
  - Priority Queue: Q.top(), Q.push(), Q.pop(), Q.empty()
  - Sets: set<key, comparison> S;
- Java, java.util

Data Structure	Abstract class	Concrete class	Methods
Stack	No interface	`Stack`	`pop`, `push`, `empty`, `peek`
Queue	`List`	`ArrayList`, `LinkedList`	`add`, `remove`, `clear`
Dictionaries	`Map`	`HashMap`, `Hashtable`	`put`, `get`, `contains`
Priority Queue	`SortedMap`	`TreeMap`	`firstKey`, `lastKey`, `headMap`
Sets	`Set`	`HashSet`	`add`, `remove`, `contains`

2.3 Program Design Example: Going to War

- 34 페이지를 읽고 오세요.
- FIFO queue를 사용해야 함.
- Card 구분
  - char values[] = "23456789TJQKA";
  - char suits[] = "cdhs";

```
4d Ks As 4h Jh 6h Jd Qs Qh 6s 6c 2c Kc 4s Ah 3h Qd 2h 7s 9s 3c 8h Kd 7h Th Td
8d 8c 9c 7c 5d 4c Js Qc 5s Ts Jc Ad 7d Kh Tc 3s 8s 2d 2s 5h 6d Ac 5c 9h 3d 9d
6d 9d 8c 4s Kc 7c 4d Tc Kd 3s 5h 2h Ks 5c 2s Qh 8d 7d 3d Ah Js Jd 4c Jh 6c Qc
9h Qd Qs 9s Ac 8h Td Jc 7s 2d 6s As 4h Ts 6h 2c Kh Th 7h 5s 9c 5d Ad 3h 8s 3c
Ah As 4c 3s 7d Jc 5h 8s Qc Kh Td 3h 5c 9h 8c Qs 3d Ks 4d Kd 6c 6s 7h Qh 3c Jd
2h 8h 7s 2c 5d 7c 2d Tc Jh Ac 9s 9c 5s Qd 4s Js 6d Kc 2s Th 8d 9d 4h Ad 6h Ts
```