알고리즘 실습

181018 - Search, Hash

오늘의 목표

- 검색 알고리즘 실습
  - 순차 검색
  - 이진 검색
  - 해쉬

Feedback

지난 과제: 정렬

- 제출: 30 / 41 (73.17%)
    - 추가제출: 2 (78.05%)
- 질문: 103개, 연구실 방문: 6명

Feedback

- 코드 카피 검사 결과
- 알고리즘을 왜 배우는가?

File 1	File 2	Lines Matched
./week6/201403499/ (56%)	./week6/201500208/ (55%)	144
./week6/201400592/ (46%)	./week6/201401317/ (45%)	103
./week6/201403499/ (42%)	./week6/201500558/ (31%)	127
./week6/201301782/ (36%)	./week6/201403499/ (39%)	101
./week6/201301782/ (36%)	./week6/201500208/ (38%)	85
./week6/201201521/ (41%)	./week6/201301782/ (31%)	80
./week6/201301782/ (29%)	./week6/201500558/ (23%)	84
./week6/201201521/ (38%)	./week6/201500208/ (30%)	58
./week6/201400588/ (33%)	./week6/201400600/ (30%)	95
./week6/201302286/ (26%)	./week6/201400600/ (29%)	72
./week6/201201874/ (40%)	./week6/201401271/ (36%)	63
./week6/201204083/ (30%)	./week6/201400600/ (27%)	82
./week6/201500208/ (25%)	./week6/201500558/ (18%)	69
./week6/201400591/ (29%)	./week6/201400600/ (26%)	87
./week6/201400591/ (28%)	./week6/201403499/ (24%)	79
./week6/201201521/ (30%)	./week6/201500558/ (18%)	57
./week6/201400344/ (17%)	./week6/201400592/ (25%)	63
./week6/201301782/ (22%)	./week6/201400344/ (16%)	60
./week6/201400591/ (26%)	./week6/201601278/ (21%)	55

순차 검색

순차 검색

- 가장 앞에서부터 검색
- 위치(인덱스)를 반환하게 하려면?

```
In [5]: def sequentialSearch(alist, item):
            pos = 0
            found = False
            count = 0

            while pos < len(alist) and not found:
                if alist[pos] == item:
                    found = True
                else:
                    pos = pos+1
                count=count+1

            return found, count

        testlist = [1, 2, 32, 8, 17, 19, 42, 13, 0]
        print(sequentialSearch(testlist, 3))
        print(sequentialSearch(testlist, 13))

(False, 9)
(True, 8)
```

순차 검색 실습

- 위치를 함께 반환

```
[82, 26, 62, 5, 52, 77, 69, 73, 41, 30]
(True, 2, 3)
(False, 10, 10)
```

```python
def sequential_search(target, item):
    # init variable
    position = 0
    found = False
    count = 0

    # search
    while position < len(target) and not found:
        if target[position] == item:
            found = True
        else:
            position = position + 1
        count = count + 1

    # return
    return found, position, count
```

```python
def main():
    import random
    test_list = random.sample(range(0, 100), 10)
    print(test_list)
    print(sequential_search(test_list, random.choice(test_list)))
    print(sequential_search(test_list, -1))
```

이진 검색

이진 검색

- 인간이 사전을 찾는 방법과 같음
- '정렬'이 된 자료에만 적용

```python
def binary_search(target, item):
    # init variable
    first = 0
    last = len(target) - 1
    found = False
    count = 0
    mid = 0

    # search
    while first <= last and not found:
        mid = (first + last) // 2
        if target[mid] == item:
            found = True
        else:
            if item < target[mid]:
                last = mid - 1
            else:
                first = mid + 1
        count = count + 1

    return found, mid, count
```

이진 검색 재귀

- 재귀를 활용할 수도 있음

```
24  def binary_search_re(target, item):
25      if len(target) == 0:
26          return False
27      else:
28          mid = len(target) // 2
29          if target[mid] == item:
30              return True
31          else:
32              if item < target[mid]:
33                  return binary_search_re(target[:mid], item)
34              else:
35                  return binary_search_re(target[mid+1:], item)
```

해쉬

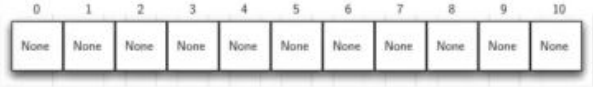
Hash

- 검색에서 O(1)의 성능을 보이는 자료구조

Hashing ?

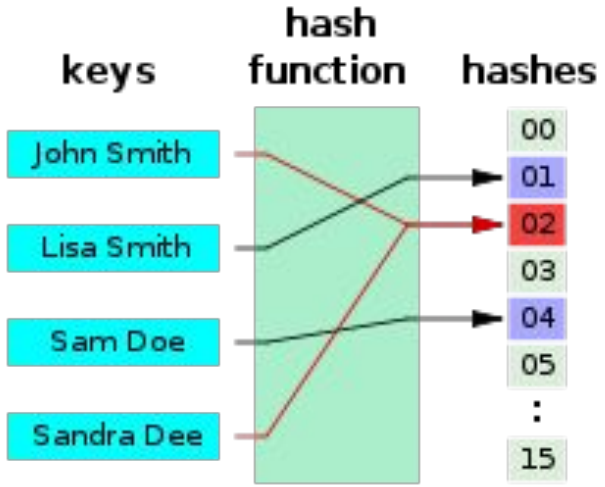
- building a data structure that can be searched in O(1) time

- Hash table
  - a collection of items which are stored in such a way as to make it easy to find them later
  - Slot
    - Each position of the hash table
  - a hash table of size m=11.



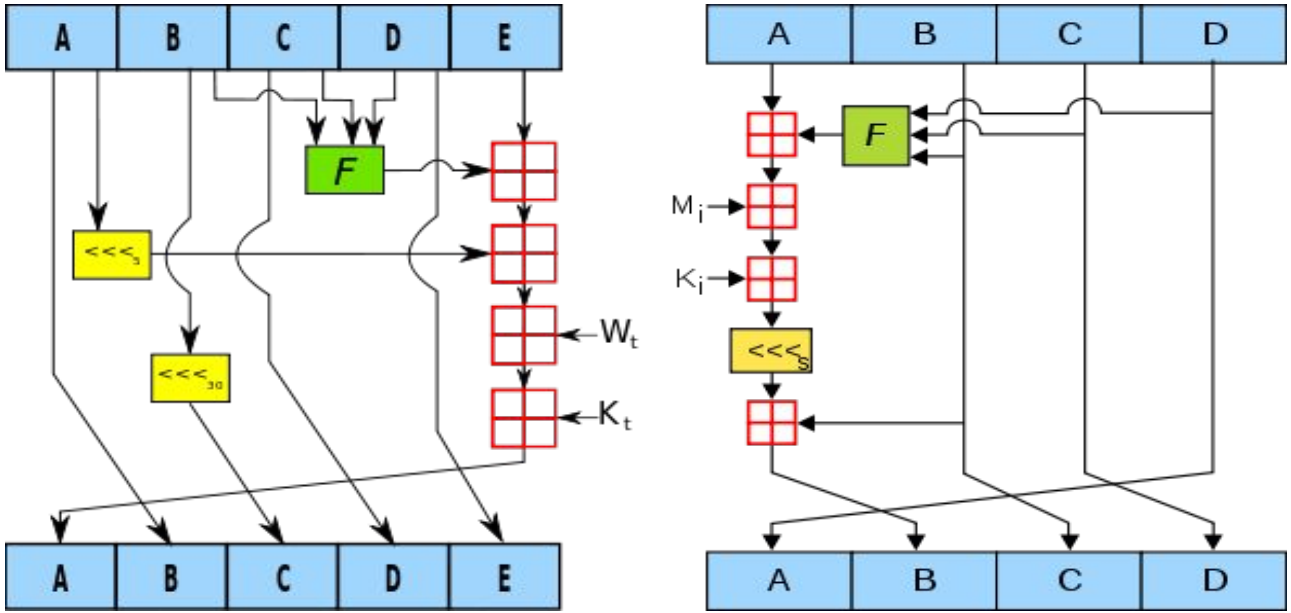
Hash Function

- Hash function
  - A hash function is any function that can be used to map data of arbitrary size to data of fixed size.
  - https://en.wikipedia.org/wiki/Hash_function
  - 입력값을 숫자로 만드는 함수



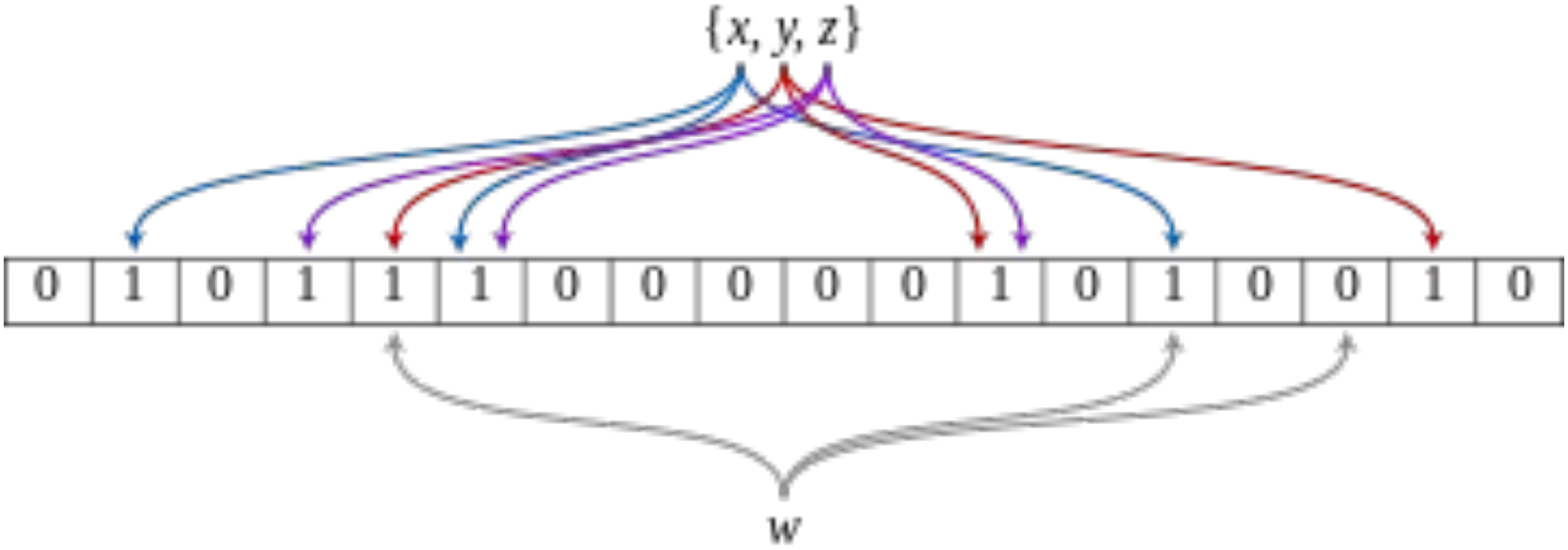
Hash Function 사용처

- 암호화 / 데이터 무결성 체크에 사용



Hash Function 사용처(2)

- Bloom filter



해쉬 클래스 실습

- 본래 ord()등을 이용하여 데이터의 key값을 계산
- 이번 실습에서는 key-data를 따로 입력
- 충돌이 일어나면 어떻게 하는가?

```
def put(self,key,data):
  hashvalue = self.hashfunction(key,len(self.slots))

  if self.slots[hashvalue] == None:
    self.slots[hashvalue] = key
    self.data[hashvalue] = data
  else:
    if self.slots[hashvalue] == key:
      self.data[hashvalue] = data  #replace
    else:
      nextslot = self.rehash(hashvalue,len(self.slots))
      while self.slots[nextslot] != None and \
                      self.slots[nextslot] != key:
        nextslot = self.rehash(nextslot,len(self.slots))

      if self.slots[nextslot] == None:
        self.slots[nextslot]=key
        self.data[nextslot]=data
      else:
        self.data[nextslot] = data #replace

def hashfunction(self,key,size):
     return key%size

def rehash(self,oldhash,size):
    return (oldhash+1)%size
```

문제)

1. slots이 가득 차면 어떻게 되는가?
    - 해결 방법은?
2. rehash(next hash key) 대신 list를 사용하여 저장하는 방법은?
    - 혹은 다른 자료형

중.간.고.사.

지난 Feedback

- 컴퓨터 프로그래밍 기본기 부족
  - 변수, 함수, 클래스, 객체, 인스턴스 정의
  - 자료형 정의
  - 할당한다. 생성(및 초기화) 한다. 의미
  - 매개변수(인자: parameter), 전달인자(인수: argument) 정의
- 잘못된 습관: 빈칸 채우기식의 코드, 과제 제출
  - 제공된 코드가 가진 의미, 사용한 기술(코드 방법) 을 모두 이해하지 않음

중간고사와 설문지

- 2018. 10. 23. 18:00 이론 강의실(405)
  - 18:00부터 시작
- 중간 강의 평가: 아직 안 열림, 열리면 입력!
- 대신 자체 설문지
  - 중간고사 이후 일정에 반영하기 위함
  - https://docs.google.com/forms/d/e/1FAIpQLSf8Ip3v2cdPYsdZM1sXFahHjFUZdzP2i_2Oc_U1mmv879Tq6w/viewform?usp=sf_link

기타 유용한 정보

실습 증명

- 2018. 10. 20.까지 (가능하면 실습시간 내 해결!)
- 순차/이진 검색, 해쉬 실습한 내용 스크린샷
- 이미지만 넣어서 업로드

출석부 및 실습 점수가 궁금하다면?

- 출석부 및 실습 채점표
  - 수업 시작 후 30분까지 지각, 이후 결석
  - 실습 딜레이 1일당: -2점
    - 딜레이 2일까지: -2
    - 이후 -1씩 추가
- 튜터의 테스트 결과

질문이 생기면?

- 이름: 문현수
- 전공: 통신및보안
- 과정: 석박사통합과정 8학기
- 연구실: 데이터네트워크연구실(공5633)
- 메일: munhyunsu@cs-cnu.org
- 알고리즘은 함께 해결해가는 과목이므로 과감하게 연락
- 이메일로 처리가 안 되는 급한일: 문자/전화 등