알고리즘 실습

181011 - Sort

오늘의 목표

- 다양한 정렬 알고리즘 학습 및 구현
  - Bubble
  - Selection
  - Insertion
  - Merge
  - Quick
- Package 제작 경험
  - 정렬 알고리즘이 담겨있는 '내 도구'를 만들자

Feedback

지난 과제: 동적 프로그래밍

- 제출: 35 / 41 (85.37%)
- 질문: 176개, 연구실 방문: 5명

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
- 어떻게 케어해주어야 하는가?

코드 검사 시작!

- 오로지 대화로 정보를 전달
- 생산자, 소비자 모두 0점 처리

손으로 풀어보자!

손으로 풀어보자!

- 9, 12, 10, 1, 5, 3, 6, 17, 7, 11
  - 과제
- 4, 19, 1, 9, 10
  - 실습

들어가기 전... python 3 sort

Sorting in Python 3

- https://docs.python.org/3/howto/sorting.html
- Python 3 리스트 정렬은 None을 반환함
- 호출한 객체(인수)를 그대로 변경
- reverse 인자 - 내림차순 정렬

```
>>> x = [4, 2, 1, 9, 6, 3]
>>> x.sort()
>>> x
[1, 2, 3, 4, 6, 9]
```

```
>>> x = [4, 2, 1, 9, 6, 3]
>>> x.sort(reverse=True)
>>> x
[9, 6, 4, 3, 2, 1]
>>>
```

Bubble Sort

Bubble Sort

- Worst O(n^2), Best O(n)
  - https://en.wikipedia.org/wiki/Bubble_sort

```
procedure bubbleSort( A : list of sortable items )
   n = length(A)
   repeat
     swapped = false
     for i = 1 to n-1 inclusive do
       /* if this pair is out of order */
       if A[i-1] > A[i] then
         /* swap them and remember something changed */
         swap( A[i-1], A[i] )
         swapped = true
       end if
     end for
   until not swapped
end procedure
```

6 5 3 1 8 7 2 4

Bubble Sort 구현

- my_utils 디렉터리에 구현한 형태
- 입력으로 넣어준 리스트를 수정

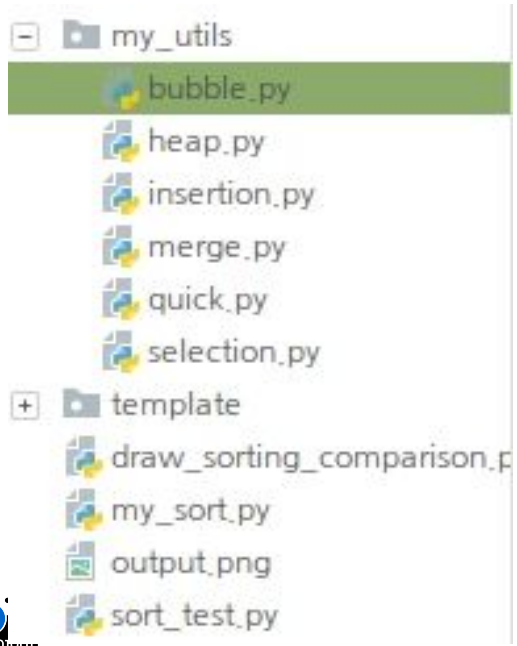
```python
def main():
    list01 = ['z', 'y', 'x', 'w', 'v',
              'u', 't', 's', 'r', 'q',
              'p', 'o', 'n', 'm', 'l',
              'k', 'j', 'i', 'h', 'g',
              'f', 'e', 'd', 'c', 'b',
              'a']
    bubble_sort(list01)
    print(list01)
    bubble_sort(list01, reverse=True)
    print(list01)


if __name__ == '__main__':
    main()
```

```python
def _swap(input_list, index_a, index_b):
    temp = input_list[index_a]
    input_list[index_a] = input_list[index_b]
    input_list[index_b] = temp


def bubble_sort(target, reverse=False):
    swapped = True
    while swapped:
        swapped = False
        for index in range(0, len(target)-1, 1):
            if reverse:
                if target[index] < target[index+1]:
                    _swap(target, index, index + 1)
                    swapped = True
            else:
                if target[index] > target[index+1]:
                    _swap(target, index, index + 1)
                    swapped = True
```

패키징



my_sort.py

```python
from my_utils.bubble import bubble_sort
from my_utils.selection import selection_sort
from my_utils.insertion import insertion_sort
from my_utils.merge import merge_sort
from my_utils.quick import quick_sort
from my_utils.heap import heap_sort
```

```python
import my_sort


LIST02 = [...]

LIST02_ASC = [...]

LIST02_DES = [...]


class SortTest(unittest.TestCase):
    def test_bubble_asc(self):
        target = copy.deepcopy(LIST02)
        my_sort.bubble_sort(target)
        self.assertEqual(target, LIST02_ASC)
```

Selection Sort

Selection Sort

- Worst O(n^2), Best O(n^2)
  - https://en.wikipedia.org/wiki/Selection_sort

```
/* a[0] to a[n-1] is the array to sort */
int i,j;
int n;

/* advance the position through the entire array */
/*   (could do j < n-1 because single element is also min element) */
for (j = 0; j < n-1; j++) {
    /* find the min element in the unsorted a[j .. n-1] */

    /* assume the min is the first element */
    int iMin = j;
    /* test against elements after j to find the smallest */
    for ( i = j+1; i < n; i++) {
        /* if this element is less, then it is the new minimum */
        if (a[i] < a[iMin]) {
            /* found new minimum; remember its index */
            iMin = i;
        }
    }

    if(iMin != j) {
        swap(a[j], a[iMin]);
    }
}
```

8
5
2
6
9
3
1
4
0
7

Insertion Sort

Insertion Sort

- Worst O(n^2), Best O(n)
  - https://en.wikipedia.org/wiki/Insertion_sort

```
for i ← 1 to length(A)-1
    j ← i
    while j > 0 and A[j-1] > A[j]
        swap A[j] and A[j-1]
        j ← j - 1
    end while
end for
```

6 5 3 1 8 7 2 4

Merge Sort

Merge Sort

- Worst Best O(n log n)
  - https://en.wikipedia.org/wiki/Merge_sort

```
function merge_sort(list m)
    // Base case. A list of zero or one elements is sorted, by definition.
    if length(m) <= 1
        return m

    // Recursive case. First, *divide* the list into equal-sized sublists.
    var list left, right
    var integer middle = length(m) / 2
    for each x in m before middle
         add x to left
    for each x in m after or equal middle
         add x to right

    // Recursively sort both sublists
    left = merge_sort(left)
    right = merge_sort(right)

    // Then merge the now-sorted sublists.
    return merge(left, right)
```

6 5 3 1 8 7 2 4

Merge Sort(2)

- Divide 한 것을 Merge

```
function merge(left, right)
    var list result
    while notempty(left) and notempty(right)
        if first(left) <= first(right)
            append first(left) to result
            left = rest(left)
        else
            append first(right) to result
            right = rest(right)
    // either left or right may have elements left
    while notempty(left)
        append first(left) to result
        left = rest(left)
    while notempty(right)
        append first(right) to result
        right = rest(right)
    return result
```

6 5 3 1 8 7 2 4

Tip) Sublist!

- Merge sort 에서는 부분 리스트(좌/우)를 추출하는 작업이 필요함

```
56      left = _merge_sort_div_asc(target[:len_target // 2])
57      right = _merge_sort_div_asc(target[len_target // 2:])
```

Tip) Python 3에서 '비공개' 표현

- 언더바(_)로 시작하는 함수/클래스는 함수/클래스 내부에서만 사용하겠다는 표시
- merge_sort(target, reverse=True)를 호출하면… 내부 동작 선택지 2개 이상
  - _merge_sort() 와 _merge()를 호출 - 오름차순, 내림차순에 따른 함수 호출
    - 함수가 많아짐, 이해가 쉬움
  - reverse에 따라 분기문만으로 분리
    - 함수 코드가 복잡해짐, 코드가 쉬움
- 이후 quick_sort()도 마찬가지

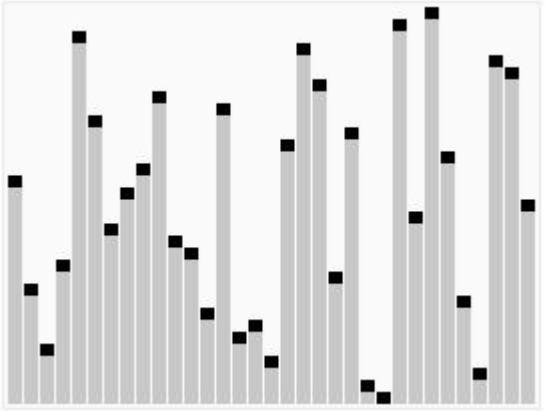
Quick Sort

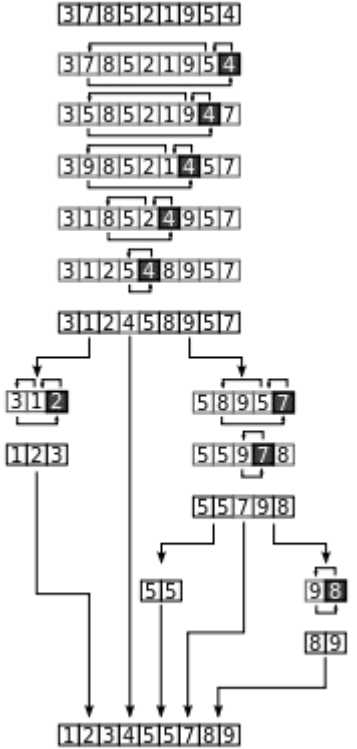
Quick Sort: Lomuto Version

- Worst(n^2) Best(n log n)
  - https://en.wikipedia.org/wiki/Quicksort

```
algorithm quicksort(A, lo, hi) is
    if lo < hi then
        p := partition(A, lo, hi)
        quicksort(A, lo, p – 1)
        quicksort(A, p + 1, hi)
```

```
algorithm partition(A, lo, hi) is
    pivot := A[hi]
    i := lo        // place for swapping
    for j := lo to hi – 1 do
        if A[j] ≤ pivot then
            swap A[i] with A[j]
            i := i + 1
    swap A[i] with A[hi]
    return i
```





Quick Sort: Hoare Versio

- Worst(n^2) Best(n log n)
  - https://en.wikipedia.org/wiki/Quicksort

```
algorithm quicksort(A, lo, hi) is
    if lo < hi then
        p := partition(A, lo, hi)
        quicksort(A, lo, p)
        quicksort(A, p + 1, hi)

algorithm partition(A, lo, hi) is
    pivot := A[lo]
    i := lo - 1
    j := hi + 1
    loop forever
        do
            i := i + 1
        while A[i] < pivot

        do
            j := j - 1
        while A[j] > pivot

        if i >= j then
            return j

        swap A[i] with A[j]
```

테스트 코드 및 성능 비교

테스트코드

- 모든 테스트가 통과되면 성공!

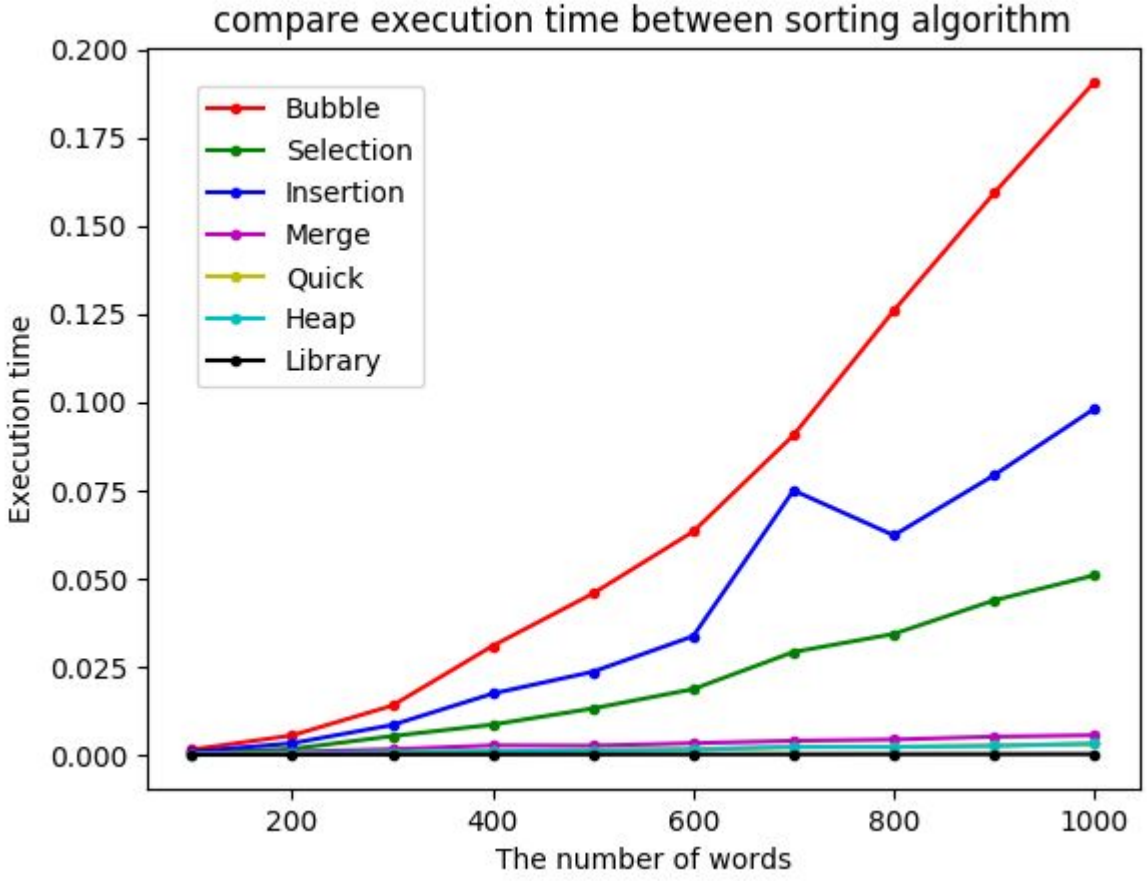
```
(venv) harny@LuHa-X1-Yoga ~/PycharmProjects/SCSC/week6 $ python3 -m unittest -vv sort_test.py
test_bubble_asc (sort_test.SortTest) ... ok
test_bubble_des (sort_test.SortTest) ... ok
test_insertion_asc (sort_test.SortTest) ... ok
test_insertion_des (sort_test.SortTest) ... ok
test_merge_asc (sort_test.SortTest) ... ok
test_merge_des (sort_test.SortTest) ... ok
test_quick_asc (sort_test.SortTest) ... ok
test_quick_des (sort_test.SortTest) ... ok
test_selection_asc (sort_test.SortTest) ... ok
test_selection_des (sort_test.SortTest) ... ok

----------------------------------------------------------------------
Ran 10 tests in 0.705s

OK
```

성능비교

- 차트 그리기
  - 구현만 하면 끝
- Heap sort는 나중에...



기타 유용한 정보

과제) 2+1개!

1. 손으로 10개 그려보기(Merge, Quick 필수! 나머지 선택)
2. Bubble, Selection, Insertion, Merge, Quick 구현
3. 성능 비교 그래프
4. (선택) 분석

실습 숙제 제출

- 숙제 제출 기한: 2018. 10. 17. 23:59:59
  - 실습 전 날
- 파일 제목: AL_학번_이름_06.zip
  - 파일 제목 다를 시 채점 안 합니다.
  - .egg 안 됨!

실습 숙제 제출할 것

- 2가지 파일을 제출
- AL_학번_이름_숙제번호.zip
  - 소스코드
    - Pycharm을 사용했을 경우 Project 디렉터리에 .idea, venv 같은 디렉터리는 제외
    - Jupyter + IPython을 사용했을 경우 ‘File - Download as’ 에서 .py 다운로드 가능
  - AL_학번_이름_숙제번호.pdf
    - 보고서는 무조건 .pdf
    - .hwp, .doc 등 채점 안 함

실습 보고서에 들어가야할 것

- 목표(할 일)
- 과제를 해결하는 방법
  - 알아야 할 것
- 과제를 해결한 방법
  - 주요 소스코드: 굳이 소스코드 전체를 붙일 필요는 없음
- 결과화면
  - 결과화면 설명(해석), 테스트코드 통과
- 보고서는 기본적으로 '내가 숙제를 했음'을 보이는 것
  - 지나치게 대충 작성하면 의심하게 됨

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