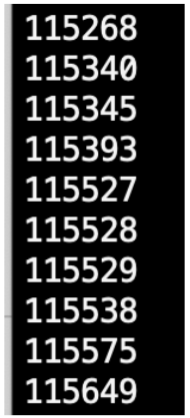
9장. 정렬 실습

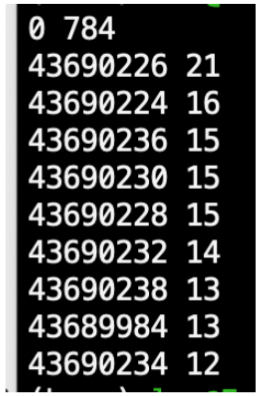
실습내용

- quicksort 를 구현하고 테스트 해본다.
- mergesort 를 구현하고 테스트 해본다.
- 다른 기준으로 정렬하는 연습을 해본다.
- 페이지 접근 횟수를 기준으로 정렬하여 접근 횟수가 많은 상위 10개의 페이지 번호를 출력한다.



linkbench_short.trc

pageSortCount.py



결과

QuickSort 구현

- QuickSort 를 구현하여 오른쪽의 코드를 실행하고 결과를 출력한다.

sort > pageSort.py > ...

```python
from quickSort import *
from mergeSort import *

def do_sort(input_file):

    data_file = open(input_file)
    A = []
    for line in data_file.readlines():
        lpn = line.split()[0]
        A.append(lpn)

    for i in range(10):
        print(A[i], end=" ")
    print("")

    quickSort(A, 0, len(A)-1)
    # mergeSort(A, 0, len(A)-1)

    for i in range(10):
        print(A[i], end=" ")
    print("")

if __name__ == "__main__":
    do_sort("linkbench_short.trc")
```

MergeSort 구현

- MergeSort 를 구현하여 오른쪽 코드를 실행하고 결과를 출력한다.

sort > pageSort.py > do_sort

```python
from quickSort import *
from mergeSort import *

def do_sort(input_file):

    data_file = open(input_file)
    A = []
    for line in data_file.readlines():
        lpn = line.split()[0]
        A.append(lpn)

    for i in range(10):
        print(A[i], end=" ")
    print("")

    # quickSort(A, 0, len(A)-1)
    mergeSort(A, 0, len(A)-1)

    for i in range(10):
        print(A[i], end=" ")
    print("")

if __name__ == "__main__":
    do_sort("linkbench_short.trc")
```

제출물

- 1) quickSort, mergeSort, pageSort 소스코드
- 2) 결과 화면 캡처
  - QuickSort 실행 및 결과
  - MergeSort 실행 및 결과

- 위의 모든 내용을 캡처하여 하나의 pdf 파일로 만들어 제출
- 다음 주 수업시간 전까지 제출