CSE 13S Winter Quarter 2022
Assignment 3: Sorting: Putting your affairs in order

Description of the Program:

The goal of this assignment is to familiarize ourselves with different sorting algorithms and to ultimately determine which sorting algorithms are the most efficient. In this assignment we will be implementing quick sort, batcher sort, heap sort, and insert sort. To observe each of their efficiencies, we will be recording each method's number of elements, moves, and comparisons.

Files to be included in the "asgn1" directory

- insert.c and insert.h
    - insert.c is where we are to implement the insert sorting algorithm, and isert.h specifies the interface to insert.c
- batcher.c and batcher.h
    - batcher.c is where we are to implement the batcher sorting algorithm and batcher.h specifies the interface to batcher.c
- heap.c and heap.h
    - heap.c is where we are to implement the heap sorting algorithm and heap.h specifies the interface to heap.c
- quick.c and quick.h
    - quick.c is where we are to implement the quic sorting algorithm and quick.h specifies the interface to quick.c
- sorting.c
    - This is the code we are supposed to write to implement all of the sorting algorithms and contains the main function
- stats.c and stats.h
    - stats.c helps display the statistics and stats.h specifies the interface to stats.c
- sets.h
    - set.h specifies the interface for the set ADT
- Makefile
    - This is used to help compile our code.
- Makefile
    - This is also given to us and is used to help compile our code.
- README.md
    - This is a file written in markdown format to explain the purpose of the code and how to build it, in addition to any errors we found.
- DESIGN.pdf
    - Gives insight into some of the processes of coming up with the code, along with the purpose of the assignment.

Pseudocode:

- Pseudocode for the sorting algorithms are shown in the assignment document

```
- def sorting():
    - seed = input("get random seed:" )
    - length = input("get array length:" )
    - A = []
    - for i in range(length):
        - a = random.randint(1, 1000)
        - A.append(a)
    - type = input("sorting algorithm:" )
    - if type == heap:
        - heap(A)
    - if type == quick:
        - quick(A)
    - if type == insert:
        - insert(A)
    - if type == batcher:
        - batcher(A)

- class stats:
    - def compare(x, y):
        - if x < y:
            - stats.comparison += 1
            - return -1
        - return 0
    - def move(x):
        - stats.move += 1
        - return x
    - def swap(x, y):
        - temp = x
        - x = y
        - y = temp
    - def reset:
        - stats.comparison = 0
        - stats.move = 0
```

Notes about Pseudocode:

- the random integers will be produced using srandom()
- stats will be produced using a statistics module code provided to us
- initial seed is given as if there is no input (13371453)
- initial array length is given to us if there is no input (100)