PDSP Assignment 4

28 October 2023, due 4 November 2023

Run the following experiments and report your results.

1. Run selection sort on `K` random lists of size `N` and compute the mean and standard deviation of the running times. Repeat this `M` times, so you should report `M` pairs of the form `(mean_run_time, std_deviation)`.
2. Run (iterative) insertion sort on `K` random lists of size `N` and compute the mean and standard deviation of the running times. Repeat this `M` times, so you should report `M` pairs of the form `(mean_run_time, std_deviation)`.
3. Implement a variant of mergesort that switches to (iterative) insertion sort when the list length is less than than `cutoff`. Run this hybrid merge-iteration sort on `K` random lists of size `N` and compute the mean and standard deviation of the running times. Repeat this `M` times, so you should report `M` pairs of the form `(mean_run_time, std_deviation)`. Try this for different values of `cutoff` below `100`, including `cutoff = 0`.
4. Implement a variant of randomized quicksort that switches to (iterative) insertion sort when the list length is less than than `cutoff`. Run this hybrid randomized-quick-iteration sort on `K` random lists of size `N` and compute the mean and standard deviation of the running times. Repeat this `M` times, so you should report `M` pairs of the form `(mean_run_time, std_deviation)`. Try this for different values of `cutoff` below `100`, including `cutoff = 0`.

Instructions

1. Submit your final code as a single Python notebook extending these instructions. However, you can run individual experiments separately before combining them into a single notebook.
2. The assignment is open ended in terms of choosing `K`, `N` and `M` for all questions and the number of different values of `cutoff` in the last two questions. However:
    - `K` should be at least `100`
    - `N` should be at least `5000` for the first two questions and at least `50000` for the last two questions
    - `M` should be at least `5`.
    - For the last two questions, use at least `5` values of `cutoff`, other than `cutoff = 0`. If the performance improves for any value of `cutoff > 0`, try to find an optimum value for `cutoff`.
3. Use the same random lists for the first two questions. Similarly use the same random lists for the last two questions.