

CMSC 3613

Programming Assignment: Quicksort

Due Date: Check the D2L calendar for the due date.

Assignment:

The task of this project is to implement Quicksort with different pivot strategies, and make a comparison for the performance of these pivot strategies and with Mergesort.

The basic program of quicksort algorithm using the left center element as the pivot (the strategy used in the lecture notes) is provided as the reference. You need to design proper program framework to accept different pivot options, and design the tests to get the average performance for evaluation. In particular, you need to:

- 1. Design and implement the following pivot options:
 - 1). using the first element as the pivot
 - 2). randomly choosing the pivot
 - 3). choosing the median of 3 randomly selected elements as the pivot
 - 4). choosing the median of the first, the center and the last element as the pivot
- 2. Design the test data to evaluate the performance of each pivot strategy. **Please note:** in the example program, the test data was randomly generated as shown in the `void GenerateArray(int* arr, int size)` function in main.cpp, so there is also some randomness in the test results. You need to consider how to handle the randomness and make a fair comparison. **Hint:**
 - 1). How to get the average performance?
 - 2). How to get the trend of growth of the performance when the input size is different?
- 3. Keep a record of the experiments. Please save the generated unsorted arrays in unsorted.dat, the sorted arrays using 4 different pivot strategies in sorted[1-4].dat, and the time used for each strategy in time[1-4].dat. So there will be 9 output files. And as mentioned in item 2 above, there will many tests, then please save each test as the same line in all the resultant files, i.e., the test result for the array at line k in unsorted.dat will also be saved at line k in the other 8 files. When you have a new test, please incrementally add the result to all the files.
- 4. Implement Mergesort and add it to the comparison, aligned with the tasks above.
- 5. Draw a graph to compare the performance of Quicksort with different pivot strategies and Mergesort: average time used v.s. the size of the input array. Please note: make a design for your tests and get the average performance for a certain input size.

Requirements:

1. Your program should follow the instructions described above in the “Assignment” section.
2. Use C++ language for implementation. The output format has been readily defined, please follow the specification.
3. There is no command line, so to execute the program for this project, you can use:

./p05

p05 should be the name of your executable (p05 should be a fixed name).

Evaluation:

This project will be evaluated according to the correctness of the various methods specified above, and secondarily according to the quality of your code. The rubric is as follows:

Categories	Weights
function design	10%
each pivot for Quicksort	10% (* 4)
Integration of Mergesort	10%
correctness	10%
report	30%

Submission:

1. Please provide a readme file, to help with the compilation and execution of your code. For example, information about your operating system and detailed command to compile your code should be included.
2. You need to submit a report by the due date on D2L. The report should include the following items:
 - 1) Your name and UCO email address
 - 2) The project number, i.e., p05
 - 3) A brief discussion of your design and implementation: just like an explanation of your idea in an interview.
 - 4) Draw a graph to compare the performance of different pivot strategies. In this graph, please include four curves showing the time used for different sizes of the input array. For example, the horizontal axis is the size of the array, and the vertical axis is the time used. Please draw the curves in the same graph for your Quicksort with four different pivot strategies and Mergesort. You can use any tool to draw the graph, e.g. use Microsoft Excel. Here is a reference to create the graph in Excel: <https://www.youtube.com/watch?v=WaNMMnDHbTl0> (maybe you can try “Scatters with smooth lines and markers”). (10%)
 - 5) Please also answer the following questions in this report:
 - Which strategy do you think is the best among the four above? Show your analysis. (4%)
 - What's the worst case for quicksort using the first element as the pivot? (4%)
 - Do you think Quicksort is faster than Insertion Sort? Please briefly justify it. (4%)
 - Some people argue that on average Mergesort could be faster than Quicksort when the input array is big. Do you agree? Please share your thoughts. (4%)
 - Can you propose a better pivot strategy for Quicksort than these given options? (4%)
 - 6) A screenshot of a test run.
3. Your source code also needs to submitted.
4. All the files need to be zipped as **p05_group*.zip**, where * means your group number, e.g., the submission from group 2 should be named as p05_group2.zip.

Notes:

1. To be considered on time, the program must be turned in by the due date.

2. Programs must reflect your knowledge and work, not others. You may ask others questions about algorithms, methods and programming style, but when you start writing code, you must work only with your group member(s).
3. No point, zero (0), will be given to any program containing compilation errors.