Mini-project Topics
OOLT

Nguyễn Nhất Hải, hainn@soict.hust.edu.vn

<u>Guidelines:</u>

- All the mini-projects must be designed and implemented in Java and by students themselves. If the teacher find out that students don't write the source code (even a part of it), the score will be 0.
- Milestones:
  - 10th week: Choosing mini-project topic, starting to study the mini-project
    - Monitor should submit the list of groups and corresponding mini-project topic by the end of the 13th week.
  - *11th, 12th, 13th: lab 9-10, discuss about miniproject design*
  - 14th, 15th and 16th weeks: Presenting your results at lab: 15 minutes for each group
- Requirements:
  - **Graphical User Interface based on Java FX**

Please choose one topic below, but you should keep the balance for the number of groups for each topic. Visual demonstration here means using visual objects/shapes to present algorithm (this is one way to present: https://www.youtube.com/watch?v=es2T6KY45cA).

1. An application to (visually) demonstrate three clustering algorithms:
   a. k-means
   b. k-nearest neighbours
   c. mean shift clustering
2. An application to (visually) demonstrate the following evolutionary algorithm:
   a. Genetic Algorithm
   b. Particle Swarm Optimization
   c. Hill Climbing
3. An application to (visually) demonstrate following meta-heuristic search algorithms:
   a. Simulated Annealing
   b. Tabu Search
   c. Artificial Bee Colony
4. An application to (visually) demonstrate following algorithms:
   a. Kruskal minimum Spanning Tree
   b. Prim Algorithm
   c. Dijkstra Shortest Path
5. An application to (visually) demonstrate following sorting algorithms
   a. Bubble sort
   b. Quick sort
   c. Heap sort
   d. Radix sort

    e. Merge sort
    f. Bucket sort
6. An application to (visually) demonstrate minimum flow cost algorithms
    a. Cycle canceling
    b. Out of kilter
    c. Minimum mean cycle canceling
7. An application to (visually) demonstrate Traveling Salesman Problem
    a. Using MST (minimum spanning tree)
    b. Using dynamic programming
    c. Naïve programing
8. An application to (visually) demonstrate maximum flow cost algorithms
    a. Linear Programing
    b. Ford-Fulkerson
    c. Edmonds-Karp
9. An application to (visually) demonstrate the random network generation problem
    a. Random network model
    b. Erdos-renyi model
    c. Watts-Strogatz model
    d. Barabasi-Albert model
10. An application to (visually) demonstrate the shortest path problem:
    a. Dijkstra
    b. Bellman-Ford
    c. A*
11. An application to (visually) demonstrate the strongly connected component problem:
    a. Kosaraju
    b. Tarjan

    Note: in this subject, students should also consider some other algorithms to generate strongly connected graphs first (then this will be the input of the two mentioned above algorithms)
12. An application to (visually) demonstrate the optimization algorithms:
    a. Ant colony optimization
    b. Particle Swarm Optimization
    c. Simulated Annealing Optimization