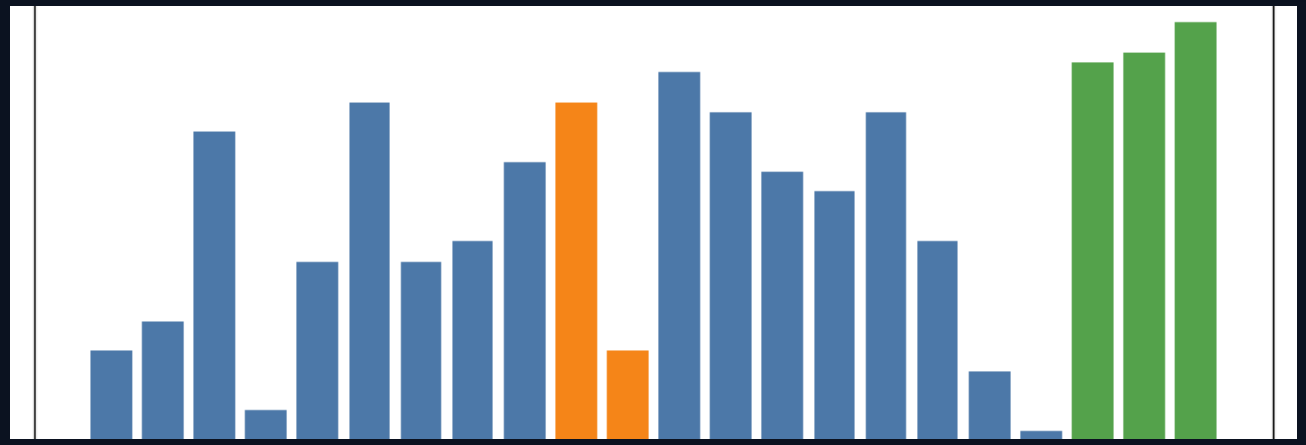
ĐỒ ÁN MÔN TRÍ TUỆ NHÂN TẠO (CS106)

Xây dựng hệ thống minh họa thuật toán cho môn: Cấu trúc dữ liệu & Giải thuật

Nhóm 13: | GVHD: Nguyễn Đình Hiển | UIT, T12/2025



1) Vấn đề cần giải quyết

Bối cảnh

- Môn DSA có nhiều khái niệm trừu tượng (cây, đồ thị, heap, DP).
- Sinh viên thường “học thuộc” thay vì hiểu luồng thực thi.
- Tài liệu tĩnh (slide/PDF) khó thể hiện quá trình từng bước.

Bài toán

- Tạo môi trường tương tác để quan sát trạng thái dữ liệu theo thời gian.
- Hỗ trợ nhập dữ liệu, chạy từng bước, tua nhanh/chậm.
- Tự sinh giải thích (explanation) và câu hỏi kiểm tra hiểu bài.

Mục tiêu hệ thống

- Kho thuật toán: sorting, searching, tree, graph, DP cơ bản.
- Trình phát hoạt hình (animation player) + timeline bước chạy.
- AI trợ giảng: gợi ý thuật toán & giải thích theo ngữ cảnh.
- Xuất báo cáo học tập (log, kết quả, thời gian).

2) Yêu cầu hệ thống

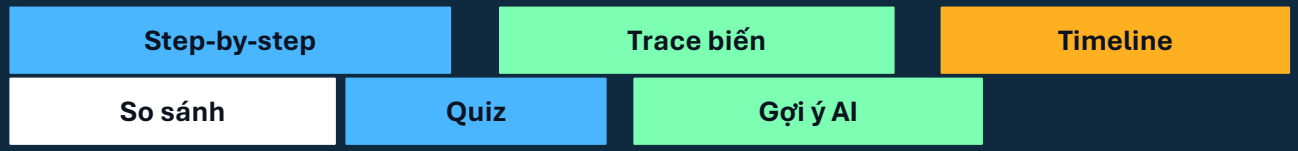
Yêu cầu chức năng (Functional)

- Chọn thuật toán và cấu trúc dữ liệu cần minh họa.
- Nhập dữ liệu (mảng, danh sách cạnh, ma trận kề, ...).
- Chạy: Step, Play/Pause, Reset, tốc độ, tua timeline.
- Hiển thị trạng thái: biến, call stack, hàng đợi/stack.
- Sinh câu hỏi trắc nghiệm/điền khuyết theo bước chạy.
- Lưu bài học: lịch sử chạy, kết quả, nhận xét.

Yêu cầu phi chức năng (Non-functional)

- Tương tác mượt (>= 30 FPS với dữ liệu vừa).
- Dễ mở rộng: thêm thuật toán mới theo plugin interface.
- Tính đúng đắn: trạng thái minh họa khớp với thuật toán.
- Khả dụng đa nền tảng (trình duyệt web).
- Logging & reproducibility: chạy lại cùng input cho cùng output.
- Bảo mật cơ bản: phân quyền tài khoản, chống input xấu.

Các chế độ minh họa



Kết hợp trực quan + tương tác + phản hồi tức thời để tăng “hiểu cơ chế” thay vì học thuộc.

3) Tổng quan giải pháp

Hệ thống gồm 3 lớp chính

Front-end (UI)

Canvas/DOM • Timeline • Input widget

- Vẽ trạng thái theo khung hình
- Điều khiển tốc độ
- Hiển thị biến & log

Engine (Core)

Step generator • State machine • Plugin API

- Chạy thuật toán và sinh “bước”
- Chuẩn hóa trạng thái
- Tách thuật toán khỏi UI

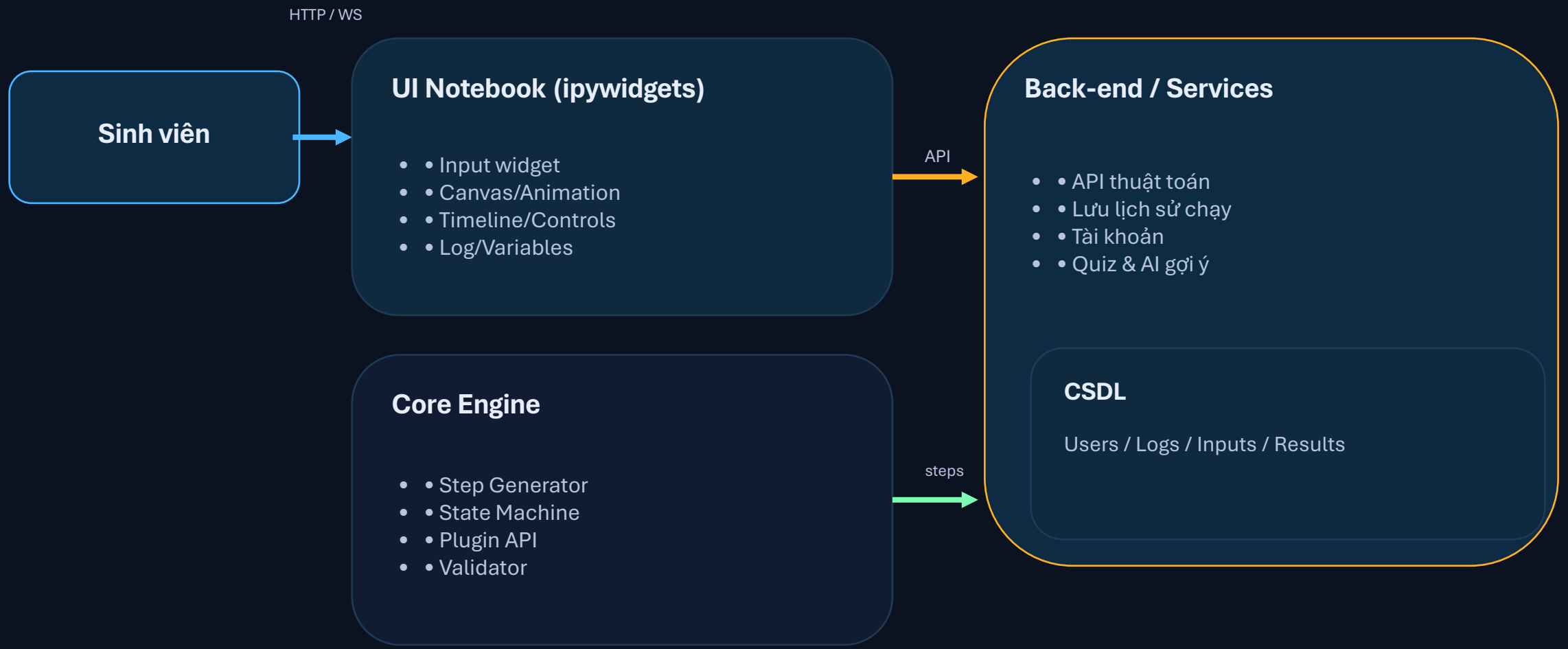
Module AI offline

Quiz • Gợi ý • Lưu log

- Sinh câu hỏi theo trace
- Gợi ý chọn thuật toán
- Lưu lịch sử học tập

4) Kiến trúc hệ thống

Sơ đồ tổng thể



5) Luồng xử lý

Từ input đến animation

Ý tưởng chính: mô hình hóa thuật toán thành chuỗi bước có cấu trúc (structured trace) để UI render độc lập.

6) Thiết kế giao diện

Thuật toán

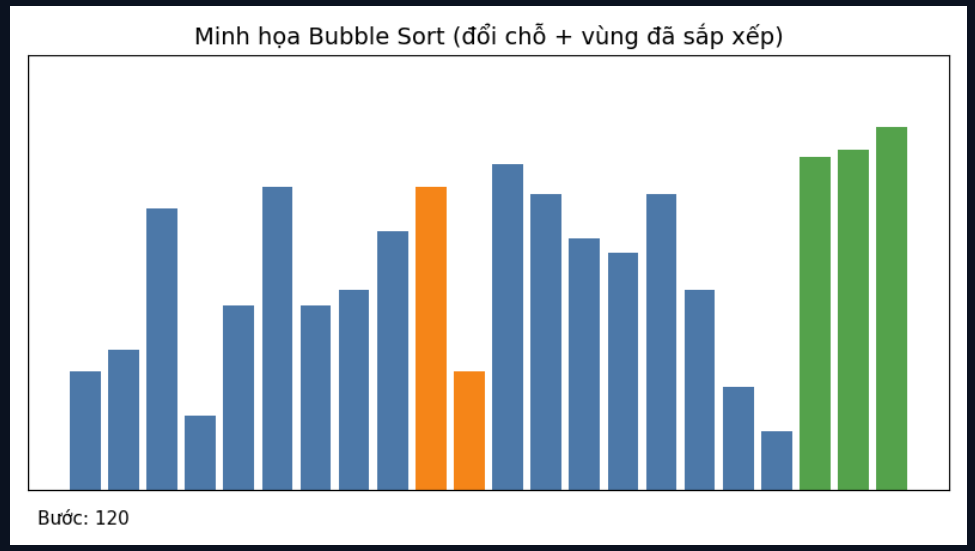
- **Sorting**
- Searching
- Tree
- Graph
- DP

Input

5 1 4 2 8 6 3

Run | Step | Reset

Canvas minh họa

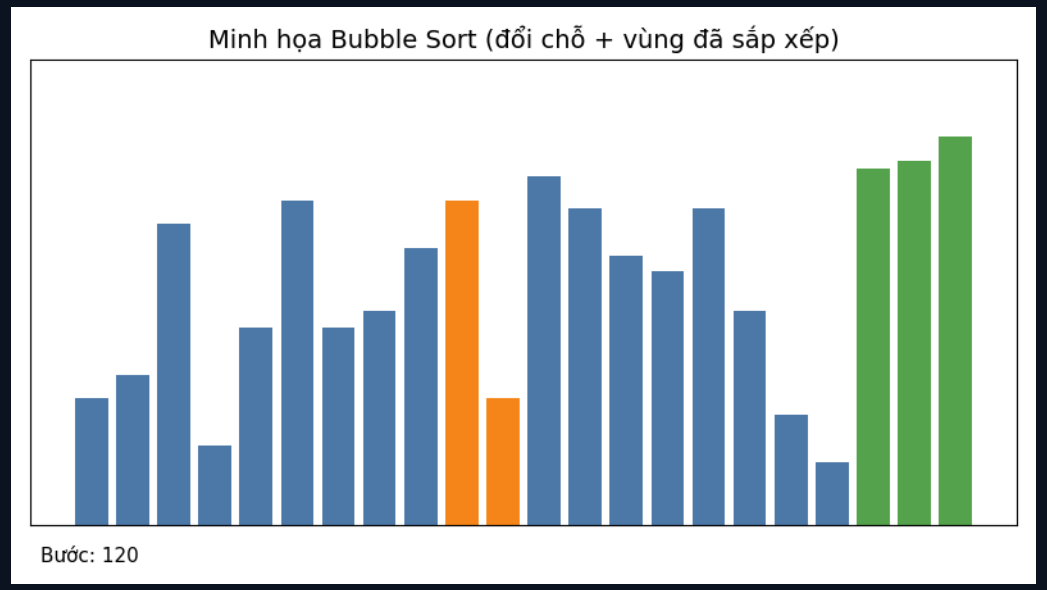


Timeline / Trace

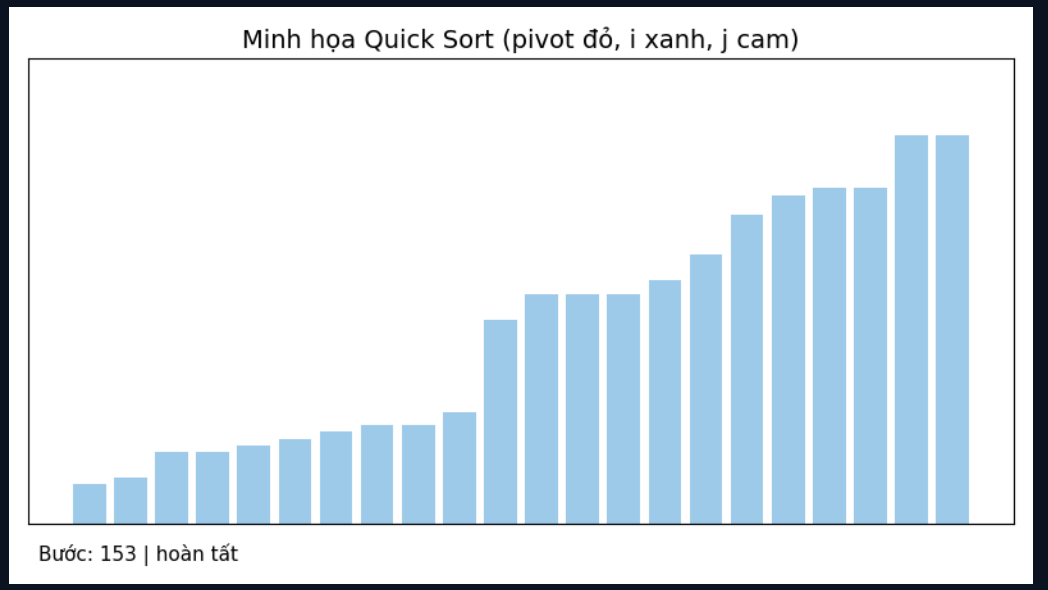
Bước 23/120 • swap(a[j], a[j+1])

7) Minh họa thuật toán: Sorting

Bubble Sort



Quick Sort

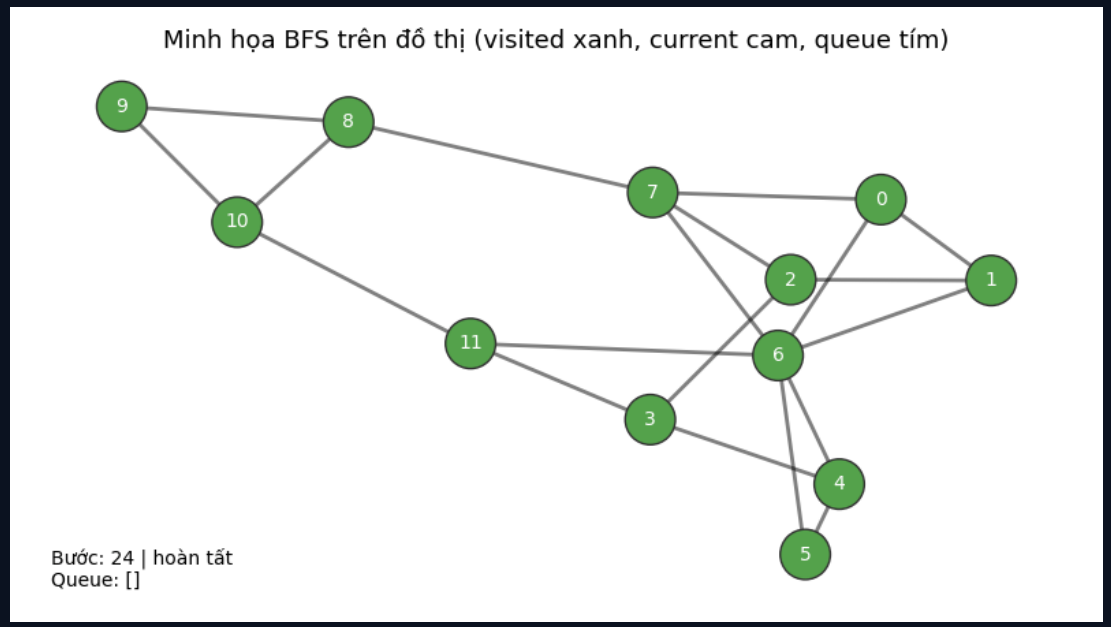


Điểm nhấn sư phạm

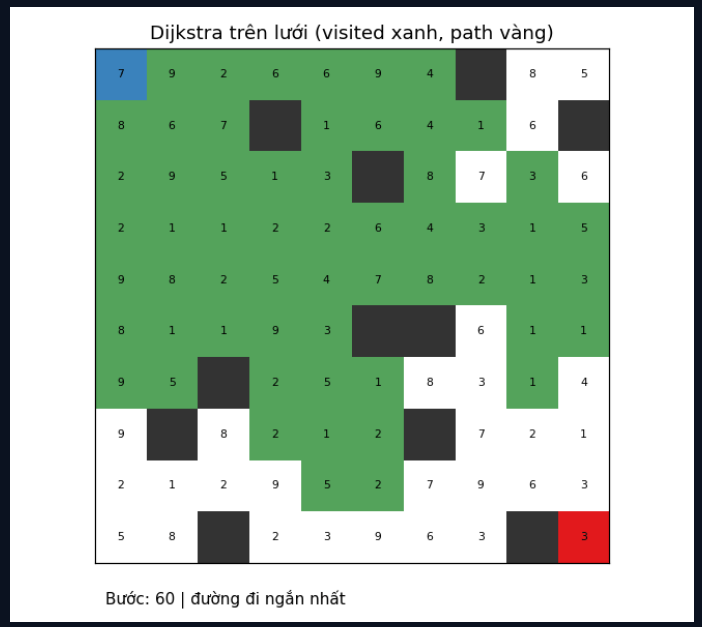
- Trực quan hóa “so sánh” vs “đổi chỗ”.
- Tô màu vùng đã sắp xếp / phân hoạch.
- Cho phép thay đổi tốc độ và dừng tại bước quan trọng.
- So sánh độ phức tạp: O(n^2) vs O(n log n) (trung bình).

8) Minh họa thuật toán: Graph

BFS (hàng đợi)



Dijkstra (priority queue)

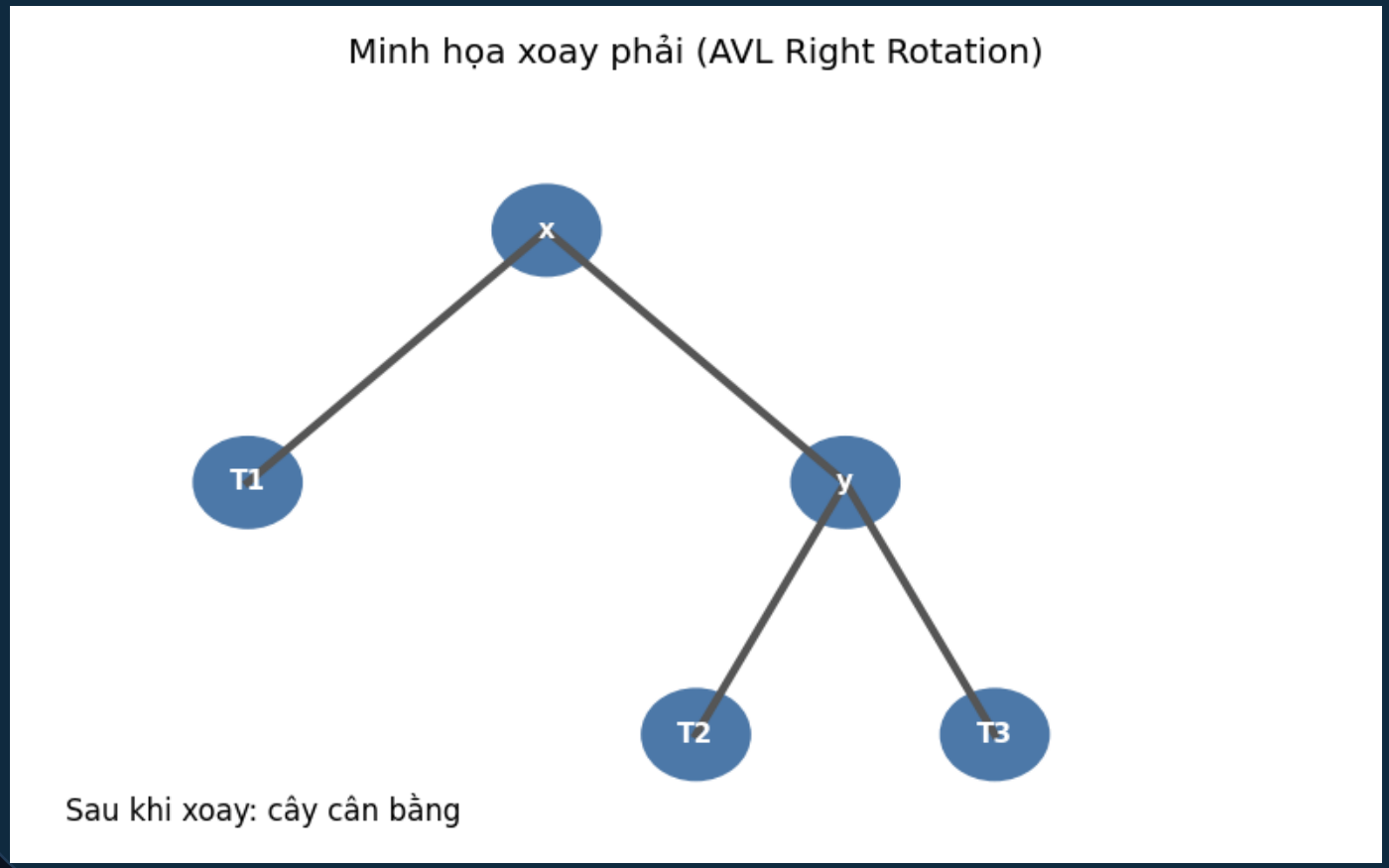


Quan sát được

- BFS: frontier (queue) và visited set.
- Dijkstra: đỉnh “chốt” theo dist nhỏ nhất + đường đi ngắn nhất.
- Hiển thị bảng dist/parent theo thời gian.

9) Minh họa cấu trúc dữ liệu: Cây

Duyệt BST giúp mô phỏng đường đi duyệt tìm kiếm nhị phân



Ý nghĩa

- Cân bằng chiều cao => tìm kiếm O(log n).
- Hiển thị BF (balance factor).
- Cho phép xem trước/sau xoay.
- Áp dụng cho insert/delete.

Mục tiêu AI (offline): hỗ trợ học tập, giải thích được
1) Hỏi đáp theo [context] (RAG-lite)
- Truy xuất đoạn liên quan từ THEORY_CONTENT bằng TF-IDF (không Internet)
- Trả lời theo template (định nghĩa / so sánh / khi nào dùng / độ phức tạp)
2) Tư vấn chọn thuật toán (Expert system)
- Luật IF-THEN theo ràng buộc: n, nearly_sorted, stable, memory, weighted/negative, ...
- Trả về (gợi ý + lý do) để dễ chấm và dễ hiểu
3) Chấm bài & phản hồi nhanh

10) Thành phần “AI” trong đồ án

Rule-based + tìm kiếm

1) Gợi ý chọn thuật toán

- Tri thức miền: bảng luật IF-THEN (dựa trên input & mục tiêu).
- Heuristic: ưu tiên thuật toán phù hợp kích thước dữ liệu.
- Giải thích được: hiển thị “vì sao gợi ý”.

2) Sinh câu hỏi (Quiz)

- Từ trace -> tạo câu hỏi theo bước then chốt.
- Ví dụ: “sau bước này, queue chứa gì?”
- Chấm điểm + phản hồi ngay.

3) Phát hiện lỗi thường gặp

- Kiểm tra input; cảnh báo trường hợp biên.
- So sánh kết quả người học dự đoán với trace chuẩn.
- Gợi ý học lại khái niệm liên quan.

4) Tự điều chỉnh độ khó

- Theo lịch sử làm quiz và thời gian hoàn thành.
- Sinh test case khó dần (tăng n, thêm trường hợp đặc biệt).
- Lộ trình học cá nhân hóa.

11) Cơ sở lý thuyết & mã giả

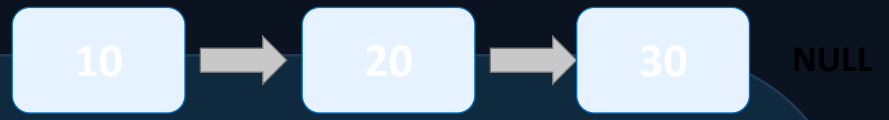
Bubble Sort (mã giả)

```
BUBBLE_SORT(A):
  n <- length(A)
  for i <- 0 .. n-1:
    swapped <- false
    for j <- 0 .. n-i-2:
      emit_state(A, compare=(j,j+1))
      if A[j] > A[j+1]:
        swap(A[j], A[j+1])
        swapped <- true
        emit_state(A, swap=(j,j+1))
    if swapped == false: break
  return A
```

Ý tưởng “emit_state”

- Engine không vẽ trực tiếp.
- Mỗi thao tác quan trọng -> phát ra một “state”.
- UI nhận state -> render frame.
- State gồm: mảng, con trỏ i/j, vùng đã sắp xếp, message.

12) Minh họa cấu trúc dữ liệu: Linked List



Singly Linked List (mã giả)

```
INSERT_HEAD(head, x):
  node <- new Node(x)
  node.next <- head
  head <- node
  emit_state(list=head, highlight=node)

DELETE_VALUE(head, x):
  dummy.next <- head
  prev <- dummy; cur <- head
  while cur != null and cur.val != x:
    prev <- cur; cur <- cur.next
  if cur != null: prev.next <- cur.next
  head <- dummy.next
  emit_state(list=head, highlight=cur)

REVERSE(head):
  prev <- null; cur <- head
  while cur != null:
    nxt <- cur.next
    cur.next <- prev
    prev <- cur
    cur <- nxt
    emit_state(list=prev, current=cur)
```

Điểm cần minh họa

- Cấu trúc Node {val, next} và liên kết giữa các nút.

Chèn/Xóa: cập nhật con trỏ next (thay đổi liên kết).
Duyệt & tìm kiếm: current di chuyển theo next.
Reverse: theo dõi prev–cur–nxt qua từng bước.
Màu sắc: node hiện tại, node bị thay đổi liên kết, NULL.

Thiết kế theo hướng plugin

```
interface AlgorithmPlugin {
  id: string
  name: string
  parseInput(raw): Input
  generator(input): Iterable<State>
  render(state): RenderModel
  quiz(state): Question[]
}

// State: snapshot tối thiểu để tái tạo hình vẽ
// RenderModel: mô tả hình học (nodes/edges/bars)
```

Lợi ích

- Thêm thuật toán mới mà không chạm UI tổng.
- Test được từng plugin (unit test).
- Có thể chạy “headless” để benchmark.
- Tái sử dụng engine cho mobile/desktop.

Nguyên tắc: “Algorithm -> Trace -> Render”

Trace là hợp đồng (contract) giữa engine và UI: dễ debug, dễ tái lập, dễ sinh quiz.

14) Phương pháp thực hiện

Quy trình đề xuất (gọn, phù hợp đồ án môn học)

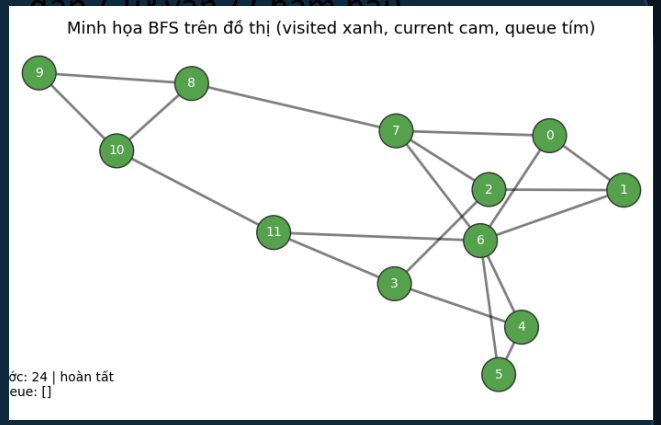
1) Phân tích yêu cầu	Use-case, thuật toán mục tiêu, input/output
2) Thiết kế	Kiến trúc, plugin API, state schema
3) Triển khai	Engine + UI + lưu log + quiz
4) Kiểm thử	Correctness, UI, hiệu năng
5) Đánh giá	Khảo sát người học, benchmark

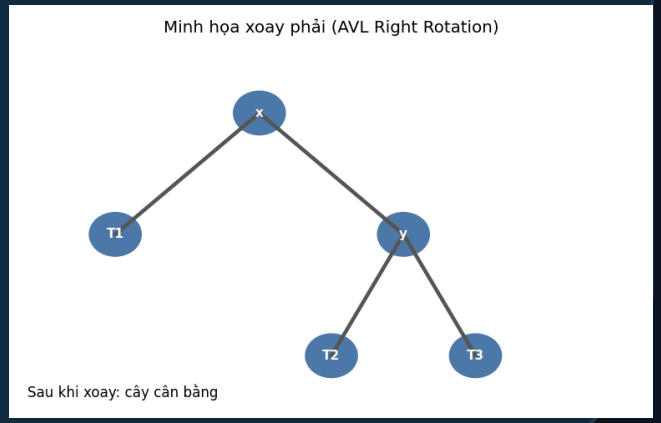
16) Demo

Video demo (MP4)

Ảnh minh họa nhanh

Demo thêm: Tab AI Trợ giảng (Hỏi đáp / Tư vấn / Chấm bài)





17) Hướng dẫn sử dụng nhanh

4 bước để học với hệ thống

1 Chọn thuật toán

Sorting/Graph/Tree...

2 Nhập dữ liệu

Mảng, danh sách cạnh...

3 Chạy & quan sát

Step/Play, xem biến

4 Làm quiz

Kiểm tra hiểu bài

Tip: dùng “Step” ở những đoạn khó (partition của Quick Sort, relax của Dijkstra, rotation của AVL).

18) Hướng phát triển

Mở rộng thuật toán

- DP: LCS, knapsack, shortest path on DAG.
- Cấu trúc dữ liệu nâng cao: segment tree, trie.
- String algorithms: KMP, Z, suffix array (cơ bản).

Trải nghiệm học tập

- Bài học theo “mission” + chấm điểm.
- Lộ trình cá nhân hóa theo điểm quiz.
- Chế độ giảng viên: tạo lớp và giao bài.

AI trợ giảng offline (tùy chọn)

- RAG trên giáo trình/slide để trả lời đúng ngữ cảnh.
- Tự sinh input “gây lỗi” để dạy case biên.
- Chẩn đoán sai lầm từ log thao tác.

Kỹ thuật

- Chạy offline (PWA) và đồng bộ khi có mạng.
- Ghi hình (record) và chia sẻ link bài học.
- Tối ưu render bằng WebGL cho dataset lớn.

19) Kết luận

DSA Visualizer cung cấp:

- Minh họa thuật toán theo từng bước + timeline.
- Tách engine và UI giúp mở rộng nhanh.
- AI trợ giảng (rule-based) để gợi ý và sinh quiz.
- Có thể dùng cho tự học hoặc trợ giảng trên lớp.

Q & A