Tổng Kết Điểm Lớp Tổ Hợp & Lý Thuyết Đồ Thị

Nguyễn Quản Bá Hồng*

Ngày 1 tháng 8 năm 2025

Mục lục

1 UMT Summer Semester 2025/1387: Combinatorics & Graph Theory

1.1 Comments on weekly reports & Final-term projects

Remark 1. *Điểm vấn đáp, phản biện đóng vai trò lớn. Dù sinh viên nộp report với nội dung kiến thức khủng (thậm chí có nhiều sinh viên sử dụng các kiến thức Toán học ngoài tầm hiểu biết hiện tại của Giảng viên có background về Toán), code siêu clean & siêu nặng OOP, nhưng không trả lời được các cây hỏi cơ bản về định nghĩa các khái niệm cơ bản, các công cụ & thuật toán mà sinh viên đó sử dụng, thì không thể chấm cao như nội dung code & report thể hiện được.*

Tiêu chí chấm điểm là sử dụng AI bao nhiêu vs. tự code bao nhiêu, với mức độ hiểu & kiểm soát các phương diện của codes mà chính bản thân sinh viên đó đã submit ở mức nào.

1. Võ Ngọc Trâm Anh.

- **Weekly reports.**
- **Final-term projects.**
  (a) Project 4, Bài toán 1: In biểu đồ Ferrers & Ferrers chuyển vị sai định dạng: phải sắp xếp theo thứ tự không tăng chứ không phải không giảm. In dấu khoảng trắng ở bên phải chứ không phải bên trái.
  - BT1 Ferrers: fixed. $\boxed{0.5}$.
  - BT2 so sánh $p_k(n), p_{\max}(n,k)$: $\boxed{0.5}$.
  - BT3 self-conjugate partition: $\boxed{1}$.
  - BT4 graph & tree representations:
  - BT 5: Phần DP good.
  - BT 6:
  - BT 7:
  - BT 8–10: $\boxed{0.75}$.
  - BT 11–13: $\boxed{0.75}$.
  - BT 14–16: $\boxed{1}$.

2. Hoàng Anh.

- **Weekly reports.**
- **Final-term projects.** Code lồng trong report khác với file code (rất nặng AIs & OOP & chứa nhiều sự phức tạp không cần thiết – excessively unnecessary complications).
  - BT1 Ferrers: Căn trái chứ không phải căn phải. Đánh số biểu đồ Ferrers chuyển vị sai: đánh số bên phải theo từng dòng chứ không phải bên dưới theo từng cột. Điểm mới: Cú pháp Pythonic của Python. Sai chính tả: ~~ferreries~~ diagram $\mapsto$ Ferrers diagram. Code theo style OOP nặng hình thức, kết quả đúng. $\boxed{0.3}$.
  - BT2 so sánh $p_k(n), p_{\max}(n,k)$: $\emptyset$ $\boxed{0}$.
  - BT3 self-conjugate partition:$\emptyset$ $\boxed{0}$.
  - BT4 graph & tree representations:

*A scientist- & creative artist wannabe, a mathematics & computer science lecturer of Department of Artificial Intelligence & Data Science (AIDS), School of Technology (SOT), UMT Trường Đại học Quản lý & Công nghệ TP.HCM, Hồ Chí Minh City, Việt Nam.
E-mail: nguyenquanbahong@gmail.com & hong.nguyenquanba@umt.edu.vn. Website: https://nqbh.github.io/. GitHub: https://github.com/NQBH.

- BT 5:
- BT 6:
- BT 7:
- BT 8–10:
- BT 11–13:
- BT 14–16:

3. Võ Huỳnh Thái Bảo.

- **Weekly reports.**
- **Final-term projects.**
  - File `README.md` khá hay: $\boxed{0.2}$.
  - BT1 Ferrers: Biểu đồ Ferrers chuyển vị trong ví dụ của report bị sai: ~~3, 3, 1, 1~~ $\mapsto 3, 2, 2, 1$. Hiểu sai đề: Đề yêu cầu nhập n, k rồi xuất ra tất cả $p_k(n)$ phân hoạch của n thành k phần chứ không phải nhập đại diện 1 phân hoạch vào. $\boxed{0.25}$.
  - BT2 so sánh $p_k(n), p_{\max}(n,k)$: Hiểu sai đề: Đề yêu cầu tính 2 số $p_k(n)$ & $p_{\max}(n,k)$ – số phân hoạch của n có phần tử lớn nhất bằng đúng k trong khi bạn lại đếm $p_{\leq k}(n)$ – số phân hoạch của n có mỗi phần tử $\leq k$, i.e., phần tử lớn nhất $\leq k$ chứ không phải bằng đúng k. $p(0,k) = 1$? $p(n,k) = 0$ if $n < 0$ or $k = 0$? Công thức DP đúng. Why "ngược lại thì $dp[i][j] = dp[i][j-1]$? Why "so sánh $p(n)$ với $\sum_{k=1}^{n} p(n,k)$? $\boxed{0.1}$.
  - BT3 self-conjugate partition: "Với ~~k~~ bất kỳ, in ra tất cả các phân hoạch tự liên hợp của n" $\mapsto$ Với n bất kỳ, in ra tất cả các phân hoạch tự liên hợp của n. $(5, 3, 1)$ không là phân hoạch tự liên hợp vì chuyển vị của nó là $(3, 2, 2, 1, 1)$. Chưa xét tính chẵn lẻ của j khi thiết lập công thức cho $dp[i][j]$.
  - BT4 graph & tree representations: Chỉ viết adjacency matrix $\leftrightarrow$ adjacency list for simple graph. $\boxed{0.1}$.
  - BT 5: Lạc đề. Vẫn chấm điểm 4 hàm `dfs, is_connected, is_tree, count_component`: $\boxed{0.25}$.
  - BT 6:
  - BT 7:
  - BT 8–10:
  - BT 11–13:
  - BT 14–16:

4. Trần Mạnh Đức.

- **Weekly reports.**
- **Final-term projects.** Chém gió, thuyết minh về mặt đẹp đẽ toán học hay.
  - BT1 Ferrers: Report trình bày tốt, chi tiết, có định nghĩa & chứng minh toán học, thậm chí có phân tích độ phức tạp thuật toán của time & space. $\boxed{0.5}$.
  - BT2 so sánh $p_k(n), p_{\max}(n,k)$: Chứng minh tốt: có phép đối hợp (involution) $f(f(x)) = x$, nắm vững kiến thức ánh xạ để vận dụng thuần thục – good job, nhưng hiểu sai đề bài lập trình: yêu cầu kiểm tra lại định lý chứ không phải sử dụng định lý để giảm task còn ít hơn $\frac{1}{2}$. Phần tính $p_{\max(n,k)}$ mới khó & là món chính.
  - BT3 self-conjugate partition: "Ký hiệu này dường như chỉ các phân hoạch tự liên hợp của n đồng thời có k thành phần." "Có vẻ đề bài muốn khám phá định lý này.": `AI thinking mode/functionality`. Phát hiện đề bài thiếu "phân biệt": good. Hiểu sai đề: đề yêu cầu kiểm tra lại tính đúng đắn của định lý Glaisher), i.e., tính cả 2 $p(n), p_{\text{do}}(n)$ chứ không phải tính 1 trong 2 nhưng code thì có tính cả 2 $\Rightarrow$ report & codes không tương thích, nhất quán. Phép biến đổi "gấp giấy": completely new. Các biểu đồ trong chứng minh format chưa đúng. 2 tên hàm bị lỗi font.
  - BT4 graph & tree representations: Chiến lược cấu trúc trung gian (intermediate representation): lạ, độc đáo nhưng sai yêu cầu bài toán là phải làm đủ tất cả.
  - BT 5: New: bidirected graph, arborescence hoặc directed spanning tree, Kirchhoff's matrix tree theorem, degree matrix, ma trận Laplacian, điều kiện tồn tại perfect matching. Exercise 1.7: Why tạo n nút nhưng chỉ duyệt nút 1 tới $n-1$? nót 0 là root? Seem so. Overall: tương đối đầy đủ & trình bày chi tiết.
  - BT 6:
  - BT 7:
  - BT 8–10: 'Trong bối cảnh khoa học máy tính được đề cập, nó được hiểu là một đồ thị có hướng đối xứng (bidirected)": sound nonhuman. Có bảng mô phỏng thuật toán rõ ràng & chi tiết.
  - BT 11–13:
  - BT 14–16:

5. Nguyễn Trung Hậu.

- **Weekly reports.** Không trả lời được các câu hỏi vấn đáp về khái niệm cơ bản.
- **Final-term projects.** Lạc đề: tìm cây khung nhỏ nhất (Minimum Spanning Tree, abbr., MST) & thuật toán Krushkal & thuật toán Prim: khá nâng cao & nằm ngoài nội dung đã học. Cần đi học nhiều hơn để nắm được giới hạn các nội dung đã học.

- BT1 Ferrers: Đúng.
- BT2 so sánh $p_k(n), p_{\max}(n,k)$: Có phần chứng minh $p_k(n) = p_{\max}(n,k)$ trong report. Cài đặt thiếu $p_k(n)$? Công thức $p_{\max}(n,k) = \text{count}(n-k,k)$ đúng. Code tính $p_{\max}(n,k)$, thiếu tính $p_k(n)$ & so sánh 2 số này để kiểm tra lại định lý.
- BT3 self-conjugate partition: "1 phân hoạch $\lambda = (\lambda_1, \ldots, \lambda_k)$ là tự liên hợp có k phần khi & chỉ khi $\lambda_1 = k$: sai, e.g., $\lambda = (3,3,1) \neq \lambda^\top = (3,2,2)$. Đúng phần lý luận so sánh: "Nói chung, hai giá trị này không bằng nhau khi xét 1 k cụ thể." Code chạy đúng.
- BT4 graph & tree representations: Hiểu sai đề: đề yêu cầu viết tất cả chương trình chuyển đổi chứ không phải sử dụng 1 dạng biểu diễn làm chuẩn, e.g., danh sách kề vì khi đó độ thử thách cài đặt về cấu trúc dữ liệu của bài toán sẽ bị mất đi. Code chỉ có chuyển đổi danh sách kề sang ma trận kề & ánh xạ kề cho đồ thị vô hướng & chuyển đổi biểu diễn cây từ danh sách kề (nhờ kế thừa từ `class Graph`) sang `parent array`.
- BT 5: Prob. 13 chỉ là bài toán liệt kê, không phải bài toán chứng minh, "Việc liệt kê là không khả thi do số lượng lớn": Sơn Tân liệt kê được nên khả thi. "Phương pháp tính toán chính xác là sử dụng Định lý ma trận cây": nhưng không thấy sử dụng để giải bài toán. Exercises 1.3 & 1.4: giải thích khá tốt.
- BT 6: Phần thuyết trình 4 phương pháp giải bài toán Tree Edit Distance tốt.
- BT 7: Thuyết trình tốt về tổng quan, thiếu phần cài đặt & giải thích chi tiết thuật toán.
- BT 8–10: Phân biệt đúng 3 loại đồ thị & có ví dụ minh thực tế để minh họa. Có env `algorithm` nice. Implement chung cho 3 loại đồ thị: đúng.
- BT 11–13: Có env `algorithm` nice. Implement chung cho 3 loại đồ thị: đúng.
- BT 14–16: Lý luận Dijkstra algorithm cho đa đồ thị & đồ thị tổng quát đúng.

6. Phạm Phước Minh Hiếu.

- **Weekly reports.**
- **Final-term projects.** Dùng phần mềm LaTeX nào hay khai báo encoding gì trong preamble mà output không sắc nét, mất nét chữ & ký hiệu, không copy được?
  - BT1 Ferrers: Đúng.
  - BT2 so sánh $p_k(n), p_{\max}(n,k)$: Có ví dụ cụ thể khi $n \in [10], k = 2$.
  - BT3 self-conjugate partition: Ví dụ phân hoạch tự liên hợp khi $n = 7$ sai. Định lý Euler phát biểu sai: ~~"bằng số phân hoạch có số phần tử lẻ"~~ $\mapsto$ "bằng số phân hoạch có các phần lẻ & phân biệt đôi một". Lý luận ~~"Do 1 phân hoạch tự liên hợp luôn gồm các số lẻ không tăng"~~: có số chẵn vẫn được. Công thức truy hồi sai. New: công thức tạo hàm sinh. Lý luận "Ràng buộc số lẻ xuất hiện là do cấu trúc đối xứng: mỗi điểm ở trên đường chéo cần đối xứng với 1 điểm dưới đường chéo, nên số phần tử ở mỗi dòng phải lẻ": sai, e.g., phân hoạch $\lambda = (2,2) = \lambda^\top$ của số 4 gồm 2 phần chẵn vẫn đối xứng nên tự liên hợp.
  - BT4 graph & tree representations: Chỉ có 3 hàm chuyển đổi cho đồ thị: `matrix2list, list2matrix, list2map` & 1 hàm cho cây `buildFNCS`.
  - BT 5: Prob. 1.3: 64 MSTs? Good.
  - BT 6: "Nếu nhãn giống nhau thì chi phí là 0, nếu khác chi phí là 1": theo mặc định của [Val21] đều là 0.
  - BT 7: Thuyết minh khá tốt.
  - BT 8–10: BT10: yếu tố "có thể vô hướng hoặc có hướng, có hoặc không có chu trình)" đều đúng cho cả 3 loại đồ thị chứ không phải đặc biệt cho đồ thị tổng quát: redundant. Code xúc tích.
  - BT 11–13: Same as BFS. Code xúc tích.
  - BT 14–16: Thiếu lý luận cho loop. Thuyết minh tốt. Code xúc tích.

7. Hoàng Quang Huy.

- **Weekly reports.** "Rất có thể có một lỗi chính tả trong đề bài.": AI-detected. Gõ LaTeX xấu dã man.
- **Final-term projects.**
  - BT1 Ferrers:
  - BT2 so sánh $p_k(n), p_{\max}(n,k)$:
  - BT3 self-conjugate partition:
  - BT4 graph & tree representations:
  - BT 5:
  - BT 6:
  - BT 7:
  - BT 8–10:
  - BT 11–13:
  - BT 14–16:

8. Phan Nguyễn Duy Kha.

- **Weekly reports.**

- **Final-term projects.** Hình thức trình bày report: good, SE-professional. Thuyết minh tốt. Case giống NNT: PNDK mạnh các thủ tục xử lý đọc xuất file, sử dụng data structures thiên về SE, bù lại cho project về phân hoạch số nguyên.
  - BT1 Ferrers: Đúng.
  - BT2 so sánh $p_k(n), p_{\max}(n,k)$: Ví dụ "$p_3(5)$ (phân hoạch thành đúng 3 phần): ~~(3, 2)~~": sai $(3,2)$ chỉ có 2 phần, nên $p_3(5) = 2$ chứ không phải 3.
  - BT3 self-conjugate partition: Thiếu phần thuyết minh trong report.
  - BT4 graph & tree representations: "Duyệt upper triangle của ma trận đối xứng để tránh xử lý duplicate edges": đã thừa nhận đồ thị vô hướng, vì đồ thị có hướng có ma trận thưởng không đối xứng. Converter 2: Matrix → Extended List: nếu chỉ duyệt upper triangle thì sẽ không thêm vào được cả `outgoing[i], incoming[i]`. New: hash set, canonical form technique. Trình bày đủ & chi tiết all converters.
  - BT 5: Chưa hoàn chỉnh – only code.
  - BT 6: Thuyết minh tốt.
  - BT 7: Thuyết minh tốt.
  - BT 8–10: Phân tích: "BFS vẫn hoạt động như trên đồ thị đơn, nhờ mảng `visited` đảm bảo không thăm lại đỉnh. Đa cạnh & khuyên chỉ làm danh sách kề có thể lặp lại phần tử, không làm thay đổi logic duyệt": good critical thinking. "Đồ thị tổng quát có thể không liên thông, gồm nhiều thành phần rời.": 3 loại đồ thị đều có thể vậy.
  - BT 11–13: "Đồ thị đa cạnh, có thể có khuyên.": sai, multigraph không có khuyên, general graph mới có. Phân tích "DFS vẫn giữ nguyên nguyên lý: duyệt sâu, không thăm lại đỉnh. Mảng visited ngăn vòng lặp do đa cạnh hoặc khuyên.": good. New: DFS forest.
  - BT 14–16: Thuyết minh tốt, có time- & space complexities. New: Có xét trường hợp nếu tìm thấy 1 cạnh có trọng số âm thì stop hẳn chương trình. Có 3 hình minh họa cho 3 loại graph. BT16: New: mixed graph, negative weights, directed cycles. Có giải thích tại sao Dijkstra's algorithm không hoạt động với trọng số âm, nếu phát hiện trong số âm thì stop & suggest Bellman–Ford, hàm `check_negative_weights` (unique: các SV khác không có): SE-very good.

9. Phạm Minh Khoa.
   - **Weekly reports.** Sử dụng AI mà không edit lại.
   - **Final-term projects.** Code đậm mùi raw non-edit AIs nhưng bù lại có comment code quá nhiều. Typo: MSVV → MSSV. Thiếu tên GV.
     - Không có code Python, chỉ có code C++ nên chia đôi điểm.
     - BT1 Ferrers: đúng. $\boxed{0.25}$.
     - BT2 so sánh $p_k(n), p_{\max}(n,k)$: Hiểu sai đề. Bài toán yêu cầu tính riêng $p_k(n)$ & $p_{\max}(n,k)$ rồi so sánh chúng để kiểm tra lại định lý $p_k(n) = p_{\max}(n,k)$ chứ không phải áp dụng định lý để chỉ tính có $p_k(n)$. Phần tính $p_{\max}(n,k)$ mới khó & là phần chính của bài toán. $\boxed{0.1}$.
     - BT3 self-conjugate partition: Hiểu sai đề. Sai kết quả. Tại sao `problems.cpp, line 21`: $n - i \geq k - 1$ là điều kiện cắt tỉa để tối ưu? Sai vì bài toán chỉ phụ thuộc vào mỗi biến n, không phụ thuộc vào biến k. $\boxed{0.1}$.
     - BT4 graph & tree representations: chỉ xét simple graph & multigraph, thiếu general graph, thiếu tree hoàn toàn. Đề bài yêu cầu xử lý tất cả cặp chuyển đổi chứ không phải chỉ nêu ra 1 cặp đại diện. $\boxed{0.1}$.
     - BT 5:
     - BT 6:
     - BT 7:
     - BT 8–10:
     - BT 11–13:
     - BT 14–16:

10. Trần Thành Lợi.
    - **Weekly reports.** $\emptyset$. 0 đ.
    - **Final-term projects.** $\emptyset$. 0 đ.

11. Lê Đức Long.
    - **Weekly reports.**
    - **Final-term projects.**
      - BT1 Ferrers: Đúng.
      - BT2 so sánh $p_k(n), p_{\max}(n,k)$: Thiết lập công thức truy hồi tốt trong report nhưng cheat ở code: 2 hàm giống y nhau, khác với công thức truy hồi thiết lập trong report.
      - BT3 self-conjugate partition:
      - BT4 graph & tree representations:
      - BT 5:

- BT 6:
- BT 7:
- BT 8–10: Thiếu đồ thị có hướng.
- BT 11–13:
- BT 14–16:

12. Huỳnh Nhật Quang.

- **Weekly reports.** Làm sai set bài tập về nhà.
- **Final-term projects.** Dư demo testcases. Có phân tích time- & space complexities cho mỗi thuật toán cho mỗi bài (thiếu giải thích & derivation cho các độ phức tạp đó). Có performance analysis: benchmark performance on Intel i5 & so sánh thuật toán thông qua 3 khía cạnh: accuracy, efficiency, scalability, có cross-validation & biết tới OEIS sequences tương ứng: khá chuyên nghiệp nhưng tiếc là output sai. Memoization không phải thuật toán mà chỉ là 1 kỹ thuật để tối ưu thuật toán, phải là đệ quy với memoization (recursion with memoization). References có nhiều tài liệu khủng, nâng cao ngoài đề cương môn học. Các phần kết luận gồm thành tựu đạt được, insight học được, ứng dụng thực tế, & hướng phát triển được thuyết minh khá hay & chi tiết.
  - BT1 Ferrers: "Đảm bảo có thể phân phối ~~đều~~ cho k phần còn lại". Có env `algorithm` đẹp. "Tối ưu vì đã sắp xếp": sắp xếp là điều kiện ràng buộc cơ bản của định nghĩa phân hoạch, chưa phải tối ưu. Lỗi font VN comment code trong report. Kết quả đúng nhưng dư test cases demo không yêu cầu, output lặp lại nhiều lần. "Phân hoạch $(3,1,1)$ conjugate $(3,2)$: sai, conjugate phải là $(3,1,1)$ & sai $F^\top$.
  - BT2 so sánh $p_k(n), p_{\max}(n,k)$: "Tuy nhiên, trong thực tế": chưa hiểu định lý về mối quan hệ với conjugate. New: thuật toán enumeration cho $p_{\max}(n,k)$. Có in ra cụ thể tất cả phân hoạch: good. Kết quả thực nghiệm $p_3(6) > p_{\max}(6,3)$ sai, phải là $p_3(6) = p_{\max}(6,3)$, mâu thuẫn với định lý $\Rightarrow$ code sai.
  - BT3 self-conjugate partition: New: "phân tích theo "hook" trong Ferrers diagram"? công thức đệ quy cho self-conjugate & điều kiện biên có đúng không? Biết thêm distinct vào odd parts. "`odd_parts_count: int` - Số phân hoạch có số phần lẻ" $\mapsto$ Số phân hoạch có các phần lẻ (số phần có thể chẵn, không nhất thiết phải lẻ). Kết quả thực nghiệm: "số phân hoạch có số phần lẻ $p_1(6) + p_3(6) + p_5(6) = 5$: hiểu sai đề, phải đếm số phân hoạch có các phần là số lẻ, không phải số phần là số lẻ. Code sai kết quả.
  - BT4 graph & tree representations:
  - BT 5:
  - BT 6:
  - BT 7:
  - BT 8–10:
  - BT 11–13:
  - BT 14–16:

13. Cao Sỹ Siêu.

- **Weekly reports.**
- **Final-term projects.**
  - BT1 Ferrers: Giải tích code C++ 2 lần, thiếu phần giải thích code Python. Bù lại report chi tiết. $\boxed{0.5}$.
  - BT2 so sánh $p_k(n), p_{\max}(n,k)$: Giải tích code C++ 2 lần, thiếu phần giải thích code Python. Bù lại report chi tiết. Công thức $p_{\max}(n,k)$ sai. So sánh ~~$p_{\max}(n,k) \le p_k(n)$~~ $\mapsto p_{\max}(n,k) = p_k(n)$. $\boxed{0.3}$.
  - BT3 self-conjugate partition:
  - BT4 graph & tree representations:
  - BT 5:
  - BT 6:
  - BT 7:
  - BT 8–10:
  - BT 11–13:
  - BT 14–16:

14. Sơn Tân.

- **Weekly reports.** Ghi nhận công lao tổ chức OLP & ICPC & các thủ tục Olympic liên quan.
- **Final-term projects.** Report 154 pages: impressive (almost as long as my lecture note on Combinatorics & Graph Theory). Nhìn chung các bài tập nặng về Toán làm chưa tốt thông qua việc output sai kết quả nhưng có cố gắng, còn cái bài tập nặng về Tin, thuật toán làm từ khá tốt đến rất tốt về chi tiết & mức độ kiểm soát code. Nhầm lẫn trong việc phân biệt giữa 3 loại đồ thị do xài nhiều phiên bản khoogn nhất quán trên mạng thay vì phiên bản thống nhất của bài giảng & tài liệu tham khảo [Sha22]. Có chú thích từng dòng code. Code khá clean, dùng struct trực tiếp tối giản thay vì dùng OOP phức tạp.

- BT1 Ferrers: Đúng, có giải thích code chi tiết. $\boxed{0.5}$
- BT2 so sánh $p_k(n), p_{\max}(n,k)$: Sai định nghĩa $p_{\max}(n,k)$: "Số phân hoạch $p_{\max}(n,k)$ là số cách biểu diễn n thành tổng của các số tự nhiên không lớn hơn k": bắt buộc số lớn nhất phải $= k$, chứ không phải $\le k$. Trường hợp cơ sở sai $P(0,k) = 0$ nếu $k \neq 0$ & $P(0,0) = 1$. Code thiếu phần chính: tính $p_{\max}(n,k)$ & so sánh $p_k(n), p_{\max}(n,k)$. Kết quả sai: quá lớn.
- BT3 self-conjugate partition: Sửa lại đề khiến sai yêu cầu bài toán. (a) Viết sai đề: ~~Chỉ ra 1 phân hoạch tự liên hợp của 1 số n có k phần~~ $\mapsto$ đếm tất cả & in ra tất cả phân hoạch tự liên hợp của 1 số n có k phần. (b) Biết thêm distinct. Ví dụ phân hoạch $\lambda = (5,5,3)$ sai, vì $\lambda^\top = (3,3,3,2,2) \neq \lambda$. Có chứng minh bằng song ánh. Nhưng sai đề: đề không yêu cầu chứng minh mà yêu cầu viết code để đếm số phân hoạch tự liên hợp có lẻ phần & phân biệt đôi một (được ST sửa lại). (c) Không có giải thích ý nghĩa theo đề tự sửa. Tính đúng đắn của công thức truy hồi cho p_k^{selfcjg}? Seem right. Kết quả code sai với bộ test $n = 6, k = 3$. Ra 0 nhưng có 1 phân hoạch tự liên hợp thỏa mãn là $\lambda = (3,2,1)$.
- BT4 graph & tree representations: Trình bày cụ thể & chi tiết cho từng loại đồ thị. Trả lời đúng khi hỏi phản biện.
- BT 5: Tốt, đặc biệt là Prob. 1.3 tự liệt kê ra toàn bộ 12 spanning tree – ~~cây bao trùm~~ $\mapsto$ cây khung (thay vì dùng some advanced AI-generated algorithms như vài bạn khác).
- BT 6: ~~Chi phí mỗi phép toán là 1.~~ $\mapsto$ Chi phí chèn & xóa bằng 1 nhưng chi phí đổi tên bằng 0. Vẫn chấp nhận. "Mã hóa cây hậu tự (post-order index)": nghe lạ tai, nice. Why "chúng tôi sẽ dùng cách đơn giản hóa bài toán"?
- BT 7: Cf. postorder traversal vs. bottom-up traversal: "Gần giống postorder nhưng trong các bài toán xử lý số liệu thường gọi là bottom-up." 2 khái niệm khác nhiều hơn thế. Giải thích & cài đặt code tốt.
- BT 8–10: BT10 thiếu khuyên (loop) khi giải thích đồ thị tổng quát, "Có thể có đỉnh cô lập": đồ thị đơn hữu hạn vẫn có thể có đỉnh cô lập.
- BT 11–13: BT12 định nghĩa phiên bản đa đồ thị theo các source trên mạng nhưng cần sử dụng định nghĩa multigraph đã được học trên lớp theo [Sha22]: multigraph có thể có cạnh lặp nhưng không có khuyên. BT13 "Trong bài toán này, đồ thị có thể có nhiều thành phần liên thông.": đồ thị đơn hay đa đồ thị vẫn có thể có nhiều thành phần liên thông.
- BT 14–16: Trọng số weight được cài đặt kiểu `int`, có thể mở rộng ra cho số thực không âm `float, double`.

15. Nguyễn Ngọc Thạch.

- **Weekly reports.**
- **Final-term projects.** Viết code systematic theo thế mạnh dev API.
  - BT1 Ferrers: Sử dụng công thức đệ quy tổng cột trước trong bảng giá trị của $p_k(n)$ thay vì công thức $p_k(n) = p_{k-1}(n-1) + p_k(n-k)$ nhưng công thức này sai, e.g., $n = 4, k = 2$, $p_4(2) = 2$ nhưng $RHS = p_1(3) + p_1(2) + p_1(1) = 1+1+1 = 3$. Vì nếu bạn chọn phần đầu tiên là i, mà các phần khác không nhỏ hơn i (tức phần đầu tiên là $\lambda_k = i$), thì cần phân hoạch $n-i$ thành $k-1$ phần mà mỗi phần $\ge i$, nhưng số $p_{k-1}(n-i)$ đếm luôn mấy phân hoạch mà mỗi phần có thể $< i$ nên sai.
  - BT2 so sánh $p_k(n), p_{\max}(n,k)$: Áp dụng công thức sai như bài 1 nhưng code đệ quy backtracking không xài công thức sai này.
  - BT3 self-conjugate partition: New: gnomon, hook. Đề bài có bẫy: số phân hoạch tự liên hợp của n bằng số phân hoạch của n thành các phần lẻ & phân biệt, tức $\sum_{k=1}^{n} p_k^{\text{selfcjg}}(n) =$ tổng số phân hoạch của n thành các phần lẻ & phân biệt, chứ không phải với mỗi k. Ý tưởng dựng lại phân hoạch tự liên hợp bằng cách lồng các hook vào nhau: nice.
  - BT4 graph & tree representations:
  - BT 5:
  - BT 6:
  - BT 7:
  - BT 8–10:
  - BT 11–13:
  - BT 14–16:

16. Phan Vĩnh Tiến.

- **Weekly reports.** Integrals of trigonometrical functions $\boxed{5}$. Có bài giải lại midterm exam chi tiết nhưng chưa đầy đủ.
- **Final-term projects.**
  - BT1 Ferrers:
  - BT2 so sánh $p_k(n), p_{\max}(n,k)$:
  - BT3 self-conjugate partition:
  - BT4 graph & tree representations:
  - BT 5:
  - BT 6:
  - BT 7:
  - BT 8–10:
  - BT 11–13:
  - BT 14–16: Xài max thay vì min trong shortest path problem.

1.2 Final grades

Student	Attendance	Weekly report	Midterm	Final-term project	Total Bonus/Minus	Final grade
Võ Ngọc Trâm Anh	7.5		11.25	10	> 76	10
Hoàng Anh	7		6.5		23.75	
Võ Huỳnh Thái Bảo	7		3.75		36.25	
Trần Mạnh Đức	3		5.75		9.25	
Nguyễn Trung Hậu	−11.25		0.75		-2.5	
Phạm Phước Minh Hiếu	7.5		4		23.25	
Hoàng Quang Huy	3.25		5.25		6.75	
Phan Nguyễn Duy Kha	-3.25		7		5.25	
Phạm Minh Khoa	−3.75		0		−3.75	
Trần Thành Lợi	−16.75	0	0	0	−16.75	−16.75
Lê Đức Long	4.25		6		18.75	
Lê Công Hoàng Phúc	6.25		4.5		9	
Huỳnh Nhật Quang	−11.5		2		−10	
Cao Sỹ Siêu	6.75		5.75		37.5	
Sơn Tân	6.75		6		22.5	
Nguyễn Ngọc Thạch	3.25		8.25		40.75	10
Phan Vĩnh Tiến	3.5		11		27.5	