TUẦN 1

2.Khi nào bạn cần chứng minh tính chính xác của thuật toán trước khi cài đặt
(0/1 Điểm)

- [x] Khi bạn xây dựng thuật toán giải bài toán hoàn toàn mới
- [x] Khi việc thử nghiệm thực tế có chi phí quá lớn hoặc khó thực hiện
- [ ] Khi bạn muốn so sánh hiệu quả thuật toán của bạn với các thuật toán khác
- [x] Khi yêu cầu đề bài bài toán là cần chứng minh

3.Một số kỹ thuật thiết kế thuật toán là
(1/1 Điểm)

- [x] Vét cạn - brute force
- [x] thuật toán tham lam
- [x] Chia để trị
- [x] Quy hoạch động
- [ ] Lập trình tuyến tính
- [ ] Phân rã hàm
- [ ] Thiết kế hướng đối tượng

4.Kiểu dữ liệu nào người ta chưa quan tâm đến cách biểu diễn cụ thể trong máy tính
(1/1 Điểm)

- ○ Kiểu dữ liệu dựng sẵn
- ○ Kiểu dữ liệu cơ sở
- ◉ Kiểu dữ liệu trừu tượng
- ○ Kiểu dữ liệu thứ cấp
- ○ Kiểu dữ liệu nguyên thủy

5.Tỉ lệ điểm thành phần là
(1/1 Điểm)

- ○ Quá trình 30%, cuối kỳ 70%
- ○ Quá trình 50%, cuối kỳ 50%
- ◉ Quá trình 40%, cuối kỳ 60%
- ○ Không có lựa chọn nào ở trên là đúng cho môn này

6.Ảnh hưởng của CTDL lên chương trình là gì? Chọn các khẳng định đúng
(1/1 Điểm)

- [ ] CTDL làm thay đổi độ chính xác của thuật toán

- [x] CTLD làm thay đổi hiệu quả của chương trình
- [x] CTDL làm thay đổi bộ nhớ của chương trình
- [x] CTDL sẽ ảnh hưởng tới việc lựa chọn thuật toán

7.Khẳng định vào sau đây là đúng nhất về chương trình máy tính
(1/1 Điểm)

- ( ) Nhiều CTDL kết hợp với 1 thuật toán
- ( ) Một CTDL kết hợp với nhiều thuật toán
- (•) Các CTDL kết hợp với các thuật toán tương ứng
- ( ) Không có khẳng định nào là đúng

8.Một số CTDL sẽ được học trong môn học là
(0/1 Điểm)

- [x] Cây nhị phân tìm kiếm
- [ ] Cơ sở dữ liệu đồ thị
- [x] Danh sách tuyến tính
- [ ] K-D tree
- [x] Cây AVL

9.Các ngôn ngữ lập trình được phép sử dụng
(1/1 Điểm)

- [x] C/C++
- [x] Pascal
- [x] Java Script
- [x] Php
- [x] C#
- [x] Python

10.Tại sao cần học CTDL&TT, chọn các đáp án đúng
(1/1 Điểm)

- [x] Để có thể xây dựng chương trình hiệu quả hơn
- [x] Để có thể xử lý được các bàn toán phức tạp và khối lượng dữ liệu lớn hơn
- [x] Để bổ trợ cho khả năng lập trình
- [ ] Để ra được trường thì phải học
- [ ] Bố/mẹ bắt phải học

11.Các mô hình đánh giá hiệu quả thuật toán nào sẽ được giới thiệu trong môn học này?

(1/1 Điểm)

- [ ] Mô hình đếm câu lệnh
- [ ] Mô hình phân tích chi ly
- [ ] Mô hình phân tích ngẫu nhiên
- [x] Mô hình RAM
- [x] Mô hình tiệm cận

TUẦN 2

1.A và B là 2 thuật toán có thời gian thực hiện trong trường hợp tồi nhất tương ứng là TA(n) = 10n^2 + 5n+6, và TB(n) = 2n^3 +n. Những kết luận nào sau đây là đúng

(1/1 Điểm)

- [ ] Thuật toán B luôn cho thời gian thực hiện tồi hơn A
- [x] Thuật toán A thường nhanh hơn B trong trường hợp tổng quát
- [x] Trong một số trường hợp, thuật toán A có thể chậm hơn thuật toán B
- [ ] Thuật toán A luôn nhanh hơn thuật toán B

2.Trong đánh giá hiệu quả về thời gian thực hiện của thuật toán trong trường hợp tồi nhất thì những khẳng định nào sau đây là đúng

(0/1 Điểm)

- [x] Quan tâm đến lệnh cơ bản = lệnh được thực hiện nhiều nhất
- [x] Mọi giả sử về dữ liệu đầu vào sẽ được tạm thời bỏ qua
- [x] Nếu trong hàm có nhiều lệnh dừng hoặc return thì quan tâm đến lệnh return nào dẫn tới thời gian thực hiện tồi nhất
- [x] Chỉ cần đếm 1 lệnh cơ bản là đủ

3.Trong bài toán đua ngựa để xác định con ngựa nhanh nhất, nhì và ba trong tổng số 50 con ngựa, với đường đua 10 làn (mỗi lần đua được 10 con) thì cần tổ chức ít nhất mấy vòng đua

(0/1 Điểm)

- ( ) 6
- (•) 7
- ( ) 8
- ( ) 9

4.Đặc điểm của mô hình RAM là

(0/1 Điểm)

- [x] Đếm số lần thực hiện các câu lệnh cơ bản
- [x] Chỉ quan tâm chính đến thời gian thực hiện - thời gian CPU
- [x] Chỉ có thể đánh giá thuật toán được mô tả bằng giả ngôn ngữ hoặc ngôn ngữ lập trình cụ thể
- [x] mất nhiều thời gian để đánh giá vì phải xét từng câu lệnh đơn lẻ

5.Tại sao cần xây dựng thuật toán hiệu quả
(0/1 Điểm)

- [x] Vì máy tính không thể tăng tốc độ bộ xử lý lên mãi được
- [x] Vì kích thước dữ liệu của bài toán mà máy tính cần xử lý ngày càng tăng
- [ ] Vì chúng ta muốn rút ngắn thời gian phải chờ đợi
- [ ] Vì chi phí xây dựng thuật toán hiệu quả sẽ rẻ hơn

6.Thời gian thực hiện của thuật toán có đặc điểm là gì? Hãy chọn các đáp án đúng
(1/1 Điểm)

- [ ] Chỉ phụ thuộc vào kích thước đầu vào
- [x] Phụ thuộc vào giá trị đầu vào cụ thể của dữ liệu
- [x] Thời gian thực hiện tồi nhất là thời gian được quan tâm nhất
- [ ] Thời gian trung bình là trung bình cộng của thời gian tồi nhất và tốt nhất

7.Ưu điểm của cách đánh giá thuật toán trực tiếp là
(1/1 Điểm)

- [x] Có thể đánh giá thuật toán ngay từ mô tả
- [x] Không phụ thuộc vào cấu hình phần cứng máy tính
- [x] Có thể tái sử dụng đánh giá cũ khi so sánh thuật toán
- [ ] Đánh giá đơn giản và dễ dàng hơn mô hình gián tiếp

8.Có các phương pháp nào để giải công thức đệ quy tổng quát
(1/1 Điểm)

- [ ] Phương pháp quy nạp
- [x] Phương pháp phân rã
- [x] Phương pháp thay thế
- [x] Định lý thợ
- [x] Cây đệ quy
- [ ] Cây nhị phân
- [ ] Phương pháp rút gọn

9.Cách đo hiệu quả thuật toán gián tiếp qua hiệu quả chương trình máy tính không đủ tin cậy khi so sánh thuật toán vì
(1/1 Điểm)

- [x] Thời gian chạy phụ thuộc nhiều vào cấu hình phần cứng
- [x] Ngôn ngữ lập trình ảnh hưởng đến hiệu quả chương trình
- [x] Kinh nghiệm của người lập trình sẽ ảnh hưởng tới hiệu quả chương trình
- [x] Việc so sánh thuật toán sẽ mất nhiều thời gian

10.Những khẳng định nào sau đây về thuật toán đệ quy là đúng

(1/1 Điểm)

- [x] Thuật toán đệ quy thường ngắn gọn hơn thuật toán dùng vòng lặp
- [x] Thuật toán đệ quy thường khó gỡ lỗi hơn
- [ ] Thuật toán đệ quy thường hiệu quả hơn vòng lặp
- [ ] Thuật toán đệ quy thường ít sử dụng bộ nhớ phụ hơn thuật toán lặp

Tuần 3:

1.Những phát biểu nào sau đây đúng với thuật toán tham lam
(1/1 Point)

- [x] Xây dựng lời giải cuối cùng bằng cách kết nối các lời giải bộ phận với nhau
- [x] Luôn chọn lời giải bộ phận nào thỏa mãn ngay và dễ thấy nhất
- [ ] Trong trường hợp tồi nhất vẫn cần vét hết không gian lời giải
- [x] Thường không cho lời giải tối ưu với các bài toán phức tạp

2.Những phát biểu nào sau đây là đúng với thuật toán nhánh và cận
(1/1 Point)

- [x] Là một cải tiến của thuật toán quay lui - back tracking
- [ ] Trong trường hợp tồi nhất vẫn sẽ vét hết toàn bộ không gian lời giải
- [x] Không gian lời giải thường được minh họa dưới dạng cây trạng thái
- [x] Quá trình tìm lời giải có thể được xem như quá trình tìm được đi trên cây trạng thái
- [x] tại mỗi nút sẽ tính toán một giới hạn - lower bound dùng để xác định xem có đi tiếp nhánh con của nó hay không

3.Có nhưng cách nào để cải thiện hiệu quả của thuật toán được cài đặt đệ quy
(0/1 Point)

- [x] dùng biến phụ để giảm số lần phải gọi đệ quy

- [x] khử đệ quy (biểu diễn lại thuật toán dùng dạng vòng lặp)
- [ ] Chia bài toán thành nhiều bài toán con nhỏ hơn nữa
- [x] dùng bảng tra để lưu các kết quả trùng lặp
- [ ] mở rộng số trường hợp cơ sở để thuật toán nhanh dừng hơn

4.Trong thuật toán tham lam để giải bài toán dây chuyền sản xuất, độ phức tạp ủa thuật toán tham làm: sắp xếp các công việc theo thứ tự giảm dần điểm thưởng, chọn công việc nào có điểm thưởng cao nhất tiếp theo nếu nó không vi phạm deadline
độ phức tạp về thời gian của thuật toán này cho đầu vào n công việc là
(0/1 Point)

- ( ) O(n)
- (•) O(nlogn)
- ( ) O(n^2)
- ( ) O(2^n)

5.Có những phương pháp nào để giải bài toán tối ưu hóa tổ hợp
(1/1 Point)

- [x] Phương pháp vét cạn
- [ ] Phương pháp lặp
- [x] Phương pháp xây dựng lời giải từng bước
- [x] Phương pháp giải xấp xỉ

6.Đâu là những đặc điểm của thuật toán dạng vét cạn
(1/1 Point)

- [x] Dựa trên năng lực tính toán của máy tính để tìm lời giải
- [x] Thường không phải là các thuật toán hiệu quả
- [ ] Thường tốn bộ nhớ hơn các thuật toán khác
- [x] Thuật toán rất dễ xây dựng
- [x] Có thể chạy với đầu vào lớn hơn nếu áp dụng thêm 1 số kỹ thuật heuristic

7.Những câu hỏi trong quiz này dễ quá với bạn

- ( ) Dễ quá, em làm trong chưa tới 1 phút
- ( ) Đủ khó để kiểm tra bài cũ
- ( ) Hơi khó cho em
- ( ) Khó quá, em chả hiểu làm thế nào

8.Thuật toán nhánh cận giải bài toán phân công công việc trong slide: best-first Branch-and-bound sẽ có thời gian thực hiện là
(0/1 Point)

- O(n^2)
- O(n^3)
- O(2^n)
- O(nlogn)

9.Những khẳng định nào sau đây về thuật toán quay lui là đúng
(0/1 Point)

- Đây chỉ là một trường hợp của thuật toán vét cạn
- trong các bước đệ quy, ta đi xây dựng lời giải toàn cục bằng cách tìm lời giải bộ phận
- thuật toán chỉ rẽ sang hướng tìm mới nếu hướng hiện tại gặp bế tắc
- thuật toán sẽ dừng sớm hơn so với phương pháp lăp thông thường
- trong thuật toán, luôn phải có bước kiểm tra hợp lệ

10.Trong thuật toán xếp hậu với bàn cờ tổng quát cỡ nxn thì độ phức tạp về thời gian sẽ là
(1/1 Point)

- O(n^2)
- O(2^n)
- O(n!)
- O(n^n)

11.Phương pháp nào hay được sử dụng để chứng minh 1 thuật toán tham lam là đúng
(1/1 Point)

- Dùng quy nạp
- Dùng đệ quy
- Dùng phản chứng
- Dùng phương pháp thế

Tuần 4

1.Trong thuật toán theo hướng tiếp cận kiểu quy hoạch động, đâu là khẳng định đúng
(1/1 Point)

- Hướng tiếp cận bottom-up nhanh hơn so với top-down do không cần gọi đệ quy

- [ ] Code cho cài đặt theo hướng top-dơn đơn giản hơn
- [ ] Hướng tiếp cận top-down ít tốn bộ nhớ hơn so với bottom-up
- [x] Tất cả các phần tử trong bảng tra cần phải được điền trong cách tiếp cận theo hướng bottom-up
- [x] Hướng tiếp cận top-down có thể không cần giải hết tất cả các bài toán con

2.Trong thuật toán sắp xếp trộn chi phí để tổng hợp lời giải các bài toán con nằm ở đâu
(1/1 Point)

- ( ) Ở việc trộn dãy
- ( ) Ở việc chia đôi dãy
- ( ) Không mất chi phí tổng hợp lời giải
- ( ) Đã được kết hợp trong bước chia dãy

3.Trong bài toán của ví dụ 5. đào vàng, nếu đầu vào là ma trận kích thước nxm thì độ phức tạp về thời gian của thuật toán quy hoạch động là
(1/1 Point)

- [ ] O(n)
- [ ] O(n^n)
- [ ] O(n+m)
- [x] O(nm)

4.Trong thuật toán nhân ma trận kích thước lớn nxn, làm thế nào để tạo ra ma trận vuông từ ma trận bất kỳ
(1/1 Point)

- [x] Bổ sung thêm các cột số 0
- [ ] Xóa bớt các hàng thừa
- [x] Bổ sung thêm các hàng số 0
- [ ] Xóa bớt các cột thừa

5.Đâu là sự khác biệt chính giữa thuật toán chia để trị và quy học động
(1/1 Point)

- [x] Các bài toán con trong quy hoạch động có thể trùng nhau
- [ ] Chia để trị nhanh hơn so với quy hoạch động
- [x] Quy hoạch động thường cần có thêm bảng tra, hoặc biến phụ
- [x] Trong quy hoạch động có thể đi trực tiếp từ bài toán con để đưa ra lời giải cho bài toán lớn hơn

- [x] Chia để trị chỉ có một cách tiếp cận theo hướng top-down

6.Trong thuật toán tìm dãy CON tăng dài nhất (Ví dụ 8), nếu đầu vào là danh sách kích thước n và áp dụng thuật toán quy hoạch động thì thời gian thực hiện trong trường hợp tồi nhất sẽ là
(1/1 Point)

- [ ] O(logn)
- [x] O(n)
- [ ] O(nlogn)
- [ ] O(n^2)

7.Trong thuật toán tìm dãy tăng dài nhất (Ví dụ 9), nếu đầu vào là danh sách kích thước n thì thời gian thực hiện của thuật toán trong trường hợp tồi nhất sẽ là
(1/1 Point)

- [ ] O(nlogn)
- [x] O(n^2)
- [ ] O(n)
- [ ] O(n!)

8.Trong thuật toán tìm cặp điểm gần nhất trong không gian Euclidean 2D, làm thế nào để cải thiện hiệu quả về thời gian
(1/1 Point)

- [x] Sắp xếp trước dãy các điểm theo trục X
- [x] Sắp xếp trước các điểm theo trục Y
- [x] Chỉ cần xét các điểm trong khoảng cách d = min(Left, Right) so với lề phân chia ở giữa
- [x] Chỉ cần xét các điểm tiềm năng trong khoiangr cách d/2 theo trục Y

9.Đâu là những khẳng định khi nói về thuật toán sắp xếp nhanh - Quick Sort
(1/1 Point)

- [x] Thời gian thực hiện trong trường hợp tồi nhất là O(n^2)
- [x] Thời gian sắp xếp phụ thuộc nhiều vào cách chọn chốt
- [x] Thường kết hợp với thuật toán sắp xếp cơ bản nếu thực hiện trên dãy ít phần tử
- [x] Bước trộn của thuật toán chia để trị nằm sẵn trong quá trình phân chia dãy
- [x] Thường chốt được chọn là trung vị

10.Trong thuật toán tìm dãy con có tổng lớn nhất (Ví dụ 7), nếu đầu vào là danh sách kích thước n thì thời gian thực hiện của thuật toán trong trường hợp tồi nhất sẽ là

(1/1 Point)

- [x] O(n)
- [ ] O(n^2)
- [ ] O(logn)
- [ ] O(nlogn)

Tuần 5:

1.Đâu là khẳng định đúng về kiểu dữ liệu con trỏ
(0/1 Point)

- [x] Con trỏ chỉ lưu trữ địa chỉ ô nhớ
- [ ] Phép cộng và trừ với kiểu con trỏ sẽ là cộng hoặc trừ địa chỉ ô nhớ
- [x] Con trỏ void về lý thuyết có thể truy cập đến bất kỳ ô nhớ nào trong RAM
- [ ] Có thể gán trực tiếp giá trị biến con trỏ bằng địa chỉ ô nhớ

2.So với mảng truyền thống thì kiểu vector trong C++ STL có đặc điểm gì nổi bật
(0/1 Point)

- [x] Không cần khai báo số lượng phần tử trước
- [x] Tự động thay đổi kích thước tùy vào nhu cầu sử dụng
- [x] Hỗ trợ một số hàm sẵn để xử lý
- [x] Thời gian thực hiện thêm hoặc xóa không phải hằng số
- [x] Kiểu phần tử của vector có thể là kiểu bất kỳ

3.Với mảng kích thước biến đổi và hệ số co giãn mảng là 1 thì khi thêm lần lượt n phần tử vào mảng ta sẽ cần co giãn mảng bao nhiêu lần
(1/1 Point)

- [ ] n
- [x] logn
- [ ] $\sqrt{n}$
- [ ] n/2
- [ ] loglogn

4.Điểm khác biệt giữa cấu trúc danh sách liên kết đơn và mảng là
(1/1 Point)

- [x] Trong danh sách liên kết đơn, mỗi phần tử phải có ít nhất 1 trường kiểu con trỏ
- [ ] Danh sách liên kết đơn có thể di chuyển theo nhiều hướng
- [x] Danh sách liên kết đơn chỉ cấp phát và lưu trữ đủ các phần tử hiện có

- [x] Thêm/xóa với sanh sách liên kết đơn chỉ cần xử lý với con trỏ, không cần di chuyển phần tử
- [ ] Danh sách liên kết đơn luôn tiết kiệm bộ nhớ hơn mảng
- [x] Muốn truy cập vào danh sách luôn phải có địa chỉ phần tử đầu tiên

5.Đâu là các thao tác cơ bản mà 1 cấu trúc dữ liệu cần phải có
(1/1 Point)

- [x] Duyệt
- [x] Tìm kiếm
- [x] Thêm
- [x] Xóa
- [x] Sắp xếp
- [x] Copy

6.Khi biểu diễn đa thức bậc n tổng quát bằng máy tính, ta nên dùng sanh sách liên kết đơn vì
(1/1 Point)

- [ ] Danh sách liên kết đơn sẽ truy cập các phần tử nhanh hơn
- [x] Đa thức khuyết thì danh sách liên kết đơn sẽ tiết kiệm bộ nhớ hơn
- [ ] Dùng mảng khi thực hiên các thao tác nhân, chia đa thức sẽ mất thêm bộ nhớ phụ
- [ ] Cài đặt bằng mảng sẽ mất thời gian hơn danh sách liên kết

7.Khi xóa phần tử trong sách liên kết đơn ta cần phải làm những gì
(1/1 Point)

- [x] Cần giải phóng bộ nhớ cấp phát động
- [x] Cần điều chỉnh con trỏ nhảy bỏ qua phần tử cần xóa
- [x] Cần tìm tới địa chỉ phần tử trước vị trí cần xóa
- [x] Cần kiểm tra xem danh sách có rỗng hay không

8.Đâu là khẳng định đúng với kiểu dữ liệu mảng
(1/1 Point)

- [x] Các phần tử trong mảng phải có cùng kiểu
- [x] Địa chỉ của phần tử đầu mảng rất quan trọng
- [x] Mảng nhiều chiều chỉ là cách nhìn logic của mảng 1 chiều trên máy tính
- [x] Các phần tử trong mảng có tính cục bộ về bộ nhớ rất cao

- [ ] Có thể tăng hoặc giảm kích thước mảng dễ dàng trong quá trình thực hiện chương trình
- [x] Các phần tử trong mảng luôn có địa chỉ liên tiếp nhau trong bộ nhớ

9.Khi thêm phần tử mới vào danh sách liên kết đơn thì việc đầu tiên cần làm là gì
(0/1 Point)

- [x] Cần cấp phát bộ nhớ động
- [x] Cần tìm vị trí chèn
- [x] Cần thay đổi liên kết của con trỏ của phần tử trước vị trí chèn
- [ ] Cần tìm địa chỉ phần tử đầu danh sách liên kết

10.Deep copy - copy sâu và shallow copy - copy nông là gì
(1/1 Point)

- [x] Deep copy tạo ra bản copy riêng (vùng nhớ riêng) của các giá trị mới
- [x] shallow copy thì bản copy cũng chỉ về vùng nhớ của bản gốc
- [x] deep copy và shallow copy liên quan đến việc copy các dữ liệu có thành phần con trỏ
- [ ] Deep copy và Shallow Copy chỉ xảy ra trong Java

Tuần 6

1.Kiểu dữ liệu thuộc loại container có đặc điểm gì
(1/1 Point)

- [x] Lưu trữ không phụ thuộc vào giá trị phần tử
- [ ] Chỉ được cài đặt bằng kiểu dữ liệu liên kết
- [x] Lấy phần tử không phụ thuộc vào giá trị phần tử
- [ ] Chỉ được cài đặt bằng mảng
- [x] Phân biệt với nhau với thứ tự lưu trữ và lấy các phần tử

2.Khẳng định nào sau đây về kiểu dữ liệu Stack là đúng (Stack trong STL và trong Java)
(1/1 Point)

- [x] Các thao tác thêm và lấy ra có thời gian O(1)
- [x] Thao tác tìm kiếm phần tử có thời gian O(n)
- [ ] Thao tác tìm kiếm phần tử cũng có thời gian O(1)
- [ ] Thao tác kiểm tra stack só lượng phần tử hiện có trong Stack có thời gian O(n)

3.Kiểu dữ liệu danh sách tuyến tính - linear list có đặc điểm gì
(1/1 Point)

- [x] Các phần tử phải tuân theo thứ tự tuyến tính
- [ ] Chỉ được thêm tại cuối và xóa tại đầu danh sách
- [ ] Các phần tử có thể có kiểu khác nhau
- [ ] Chỉ có thể truy cập các phần tử một cách tuần tự

4.Trong thuật toán chuyển biểu thức dạng trung tố sang dạng hậu tố dùng Stack thì khẳng định nào sau đây là đúng
(1/1 Point)

- [ ] Stack dùng để chưa các toán hạng
- [ ] Stack dùng để chứa toán tử
- [x] Stack dùng để chưa toán tử và dấu (
- [ ] Stack dùng để chứa cả toán tử và toán hạng và dấu (

5.Khẳng định nào về kiểu dữ liệu Ngăn xếp - stack là đúng
(1/1 Point)

- [x] Đây là 1 loại container dạng LIFO
- [ ] Đây là 1 loại container dạng FIFO
- [ ] Việc thêm và xóa có thể diễn ra tại các đầu khác nhau
- [x] Phần tử thêm vào sau cùng luôn lấy ra trước tiên

6.Cho biểu thức dạng hậu tố sau, giá trị của biểu thức tính được sẽ là (các toán tử là 2 ngôi)

4 5 3 + + * 3 - 1 + + 5
(1/1 Point)

- [x] Biểu thức bị sai
- [ ] 47
- [ ] 52
- [ ] 53

7.Biểu thức dạng hậu tố có ưu điểm gì so với dạng trung tố
(1/1 Point)

- [x] Chỉ có một cách tính giá trị duy nhất
- [x] Không cần xét độ ưu tiên toán tử và dấu ngoặc
- [ ] Khi tính giá trị biểu thức, chỉ cần duyệt 1 lần
- [x] Toán tử nào gặp trước sẽ là toán tử được thực hiện đầu tiên

8.Đâu là điểm khác biệt của danh sách liên kết đơn nối vòng so với danh sách liên kết đơn thông thường

(1/1 Point)

- [x] Con trỏ next của phần tử cuối cùng được trỏ ngược về phần tử đầu
- [x] có thể quay lại phần tử ngay trước nếu đi đủ 1 vòng
- [ ] Tìm kiếm phần tử nhanh hơn so với danh sách liên kết thông thường
- [x] Việc thêm và xóa phần tử phức tạp hơn khi phải check vòng

9.Điểm khác biệt của danh sách liên kết đôi so với danh sách liên kết đơn là
(1/1 Point)

- [x] Có thể di chuyển theo 2 chiều
- [x] Tốn nhiều bộ nhớ hơn
- [x] Thêm và xóa phần tử nhanh hơn
- [ ] Tìm kiếm phần tử nhanh hơn
- [x] Cài đặt phức tạp hơn

10.Cho biểu thức dạng trung tố sau, dạng hâu tố tương ứng sẽ là gì

(3+a-b)+5-2+6/c
(1/1 Point)

- [ ] 3ab-+52-+6c/+
- [ ] 3ab-+5+2-6c/+
- [ ] 3a+b-52-+6/+
- [x] 3a+b-5+2-6c/+

TUẦN 7

1.Có những container nào có thể dùng để cài đặt hiệu quả hàng đợi ưu tiên
(1/1 Point)

- [ ] mảng vòng
- [ ] danh sách liên kết đôi
- [ ] deque
- [x] heap

2.Trong cài đặt hàng đợi dùng danh sách móc nối, việc sử dụng thêm con trỏ rear - trỏ vào cuối danh sách có tác dụng là
(1/1 Point)

- [x] Thêm phần tử vào cuối danh sách nhanh hơn
- [ ] Đếm số phần tử trong danh sách nhanh hơn

- [ ] Để phân biệt khi hàng đợi đầy và rỗng dễ dàng hơn
- [ ] Để lấy phần tử khỏi hàng đợi dễ dàng hơn

3.Mảng dạng "vòng" là mảng có đặc điểm gì
(1/1 Point)

- [x] Bộ nhớ vật lý trên máy là mảng 1 chiều
- [ ] Bộ nhớ vật lý trên máy là mảng 2 chiều
- [ ] Bộ nhớ vật lý trên máy tạo thành hình vòng tròn
- [x] Duyệt hết phần tử cuối sẽ quay ngược về đầu mảng

4.Trong thuật toán trộng k dãy n phần tử đã có thứ tự tăng dần để thu được dãy cũng có thứ tự bằng cách dùng priority queue, thuật toán có đặc điểm gì
(1/1 Point)

- [ ] dùng queue dạng max-heap
- [x] dùng queue dạng min-heap
- [x] thời gian thực hiện của thuật toán cỡ O(nk)
- [x] mỗi phần tử trong k dãy chỉ được xét 1 lần duy nhất

5.Hàng đợi 2 đầu - deque là cấu trúc dữ liệu có đặc điểm
(1/1 Point)

- [x] có thể thêm vào và lấy ra ở cả 2 đầu
- [ ] có thể được cài đặt dùng danh sách liên kết đơn
- [x] có thể được cài đặt dùng mảng dạng vòng
- [x] có thể truy cập các phần tử theo thứ tự xuôi và ngược đều được

6.Trong thuật toán VD7. Tìm quãng đường từ ô hiện tại tới ô giá trị 1 gần nhất theo hàng hoặc cột trong ma trận nhị phân.
Đọ phức tạp tính toán của thuật toán đề xuất khi áp dụng trên ma trận kích thước nxn là
(1/1 Point)

- [x] O(n^2)
- [ ] O(n^3)
- [ ] O(nlogn)
- [ ] O(n)

7.Trong bài toán tìm phần tử lớn nhất thứ k, ta sẽ dùng thêm hàng đợi ưu tiên loại nào
(1/1 Point)

- [ ] dạng max-heap

- [x] dạng min-heap
- [ ] dùng loại nào cũng được

8.Hàng đợi ưu tiên - priority queue nếu cài đặt dùng mảng thì thời gian thực hiện các thao tác thêm và lấy ra tồi nhất sẽ là
(1/1 Point)

- [x] O(n)
- [ ] O(logn)
- [ ] O(nlogn)
- [ ] O(1)

9.Làm thế nào để có thể mô phỏng các thao tác của 1 stack dùng 2 hàng đợi? Khẳng định nào là đúng
(1/1 Point)

- [ ] Không thể mô phỏng được vì hàng đợi không thể đảo được thứ tự
- [x] vẫn mô phỏng được, tuy nhiên mỗi lần lấy ra sẽ có chi phí O(n)
- [x] Mỗi lần thêm vào ta sẽ chỉ thêm vào hàng đợi đang có phần tử, hàng đợi còn lại dùng để hỗ trợ lấy ra

10.Cài đặt hàng đợi dùng mảng có đặc điểm gì
(1/1 Point)

- [x] Phải biết trước số lượng phần tử tối đa nếu dùng mảng cấp phát 1 lần
- [ ] Nhìn chung ít lãng phí bộ nhớ hơn so với danh sách liên kết
- [x] Thời gian thực hiện các thao tác thêm hoặc xóa có thể lên tới O(n) nếu dùng mảng kích thước biến đổi
- [ ] Thời gian thực hiện nhanh hơn so với cài đặt dùng danh sách liên kết

Tuần 8

Để in ra các nút trên cây theo thứ tự lần lượt nút gốc, sau đó tới các nút gần gốc trước, sau đó tới các nút xa hơn thì thuật toán duyệt cây đó ta cần dùng thêm cấu trúc dữ liệu phụ nào
(1/1 Point)

- [ ] ngăn xếp
- [x] hàng đợi
- [ ] gọi đệ quy, không cần dùng thêm cấu trúc dữ liệu phụ
- [ ] dùng vòng lặp và không cần dùng thêm cấu trúc dữ liệu phụ

Trong các phương án biểu diễn cây tổng quát, giả sử ta biết trước số lượng nút thì phương án nào tiết kiệm bộ nhớ nhất: Biển diễn qua danh nhãn nút cha, biểu diễn qua danh sách nút con, biểu diễn qua con đầu tiên và anh chị em kế tiếp
(1/1 Point)

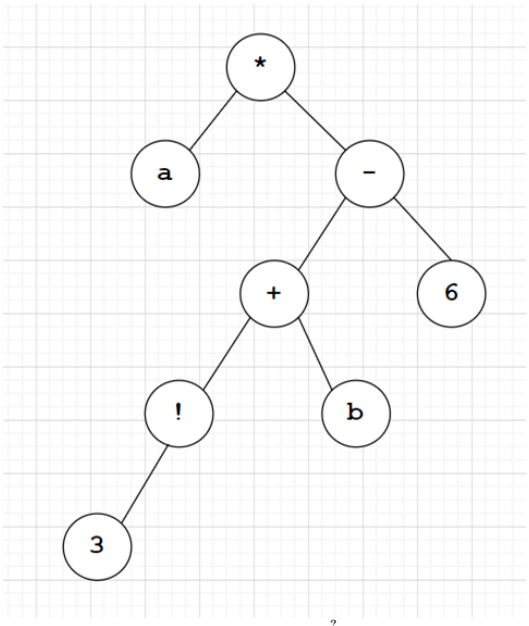
- [x] Biển diễn qua danh sách nhãn nút cha
- [ ] biểu diễn qua danh sách nút con
- [ ] biểu diễn qua con đầu tiên và anh chị em kế tiếp
- [ ] Các phương án đều tiết kiệm bộ nhớ như nhau

Chiều cao/độ sâu của của cây được tính như thế nào
(0/1 Point)

- [x] là đường đi lớn nhất tới lá
- [x] là độ sâu của nút lá lớn nhất
- [ ] là đường đi tới lá nhanh nhất
- [x] là đường đi tới nút lá ở xa gốc nhất

Cho một cây nhị phân chỉ gồm nút trong có 2 con và nút lá. Biết thứ tự duyệt cây theo
Thứ tự trước: E D B C A F G
Thứ tự giữa: B D C E F A G
Hỏi thứ tự nào sau đây là thứ tự duyệt sau đúng với cây trên?
(1/1 Point)

- [x] B C D F G A E
- [ ] D C F G A B E
- [ ] D B C F G A E
- [ ] C D B F A G E



cho cây sau, đâu là biểu thức trung tố tương ứng
(1/1 Point)

- [x] a*((3!+b)-6)
- [ ] a*(3!+b)-6
- [ ] a*3!+b-6
- [ ] a*((3+b)!-6)

Để tạo ra một cây nhị phân đầy đủ chiều cao h, số lượng nút cần phải có là bao nhiêu
(1/1 Point)

- [ ] 2^h
- [x] 2^(h+1)-1
- [ ] 2*h+1
- [ ] 2*(h+1)

trong khi duyệt cây, dữ liệu tại gốc được xử lý sau khi xử lý hết tại các nút con của nó. Thuật toán duyệt này được gọi là
(1/1 Point)

- ( ) duyệt theo thứ tự trước
- (•) duyệt theo thứ tự sau
- ( ) duyệt tuần tự
- ( ) duyệt theo thứ tự giữa

- duyệt quay lui

Có những loại nút nào trên cây
(1/1 Point)
- [x] nút gốc
- [ ] nút giữa
- [x] nút lá
- [ ] nút rìa
- [x] nút trong
- [ ] nút đỉnh

Có những kiểu quan hệ nào giữa các nút trên cây
(1/1 Point)
- [x] quan hệ ngang hàng
- [ ] quan hệ bác - cháu
- [x] quan hệ cha-con
- [x] quan hệ tổ tiên-con cháu
- [ ] quan hệ anh chị em họ hàng

Trong duyệt cây tổng quát, nếu nút lá luôn được xử lý đầu tiên thì quá trình duyệt này có thể là
(0/1 Point)
- [ ] duyệt theo thứ tự trước
- [x] duyệt theo thứ tự sau
- [ ] duyệt theo mức
- [x] duyệt theo thứ tự giữa

TUẦN 9

1.Nút lá có giá lớn nhất trên cây nhị phân tìm kiếm có đặc điểm gì
(0/1 Point)
- [ ] Chỉ nằm về cây con phải của gốc
- [x] Có thể nằm trên cây con trái của gốc
- [x] Có thể là nút gốc
- [ ] Có thể không phải là duy nhất

2.Làm thế nào để loại bỏ các số nguyên bị trùng trong danh sách n phần tử một cách hiệu quả
(0/1 Point)

- [x] sắp xếp danh sách, sau đó duyệt tuần tự, kiểm tra 2 phần tử liên tiếp có bằng nhau
- [ ] Không cần sắp xếp, tìm kiếm tuần tự từng phần tử trong danh sách, phần tử bị trùng thì sẽ tìm thấy trong phần còn lại
- [x] sắp xếp danh sách, sau đó áp dụng tìm kiếm nhị phân để tìm phần tử bị lặp
- [ ] Tìm kiếm nhị phân ngay trên danh sách ban đầu

3.Đâu là khẳng định đúng về thuật toán tìm kiếm nhị phân
(1/1 Point)

- [x] Thời gian tìm kiếm nhanh (so với các thuật toán tìm kiếm dựa trên so sánh)
- [x] Có thể thực hiện tìm kiếm trên danh sách tới cỡ hàng triệu phần tử
- [ ] Danh sách không cần có thứ tự
- [x] Sau mỗi phép so sánh, danh sách tìm kiếm giảm đi 1 nửa

4.Chiều cao lớn nhất của cây nhị phân tìm kiếm tạo từ danh sách n phần tử là
(1/1 Point)

- [ ] logn
- [ ] logn - 1
- [x] n-1
- [ ] n/2

5.Cho 1 dãy các số dương đã có thứ tự nhưng bị xen lẫn bởi các số 0, hãy xây dựng thuật toán hiệu quả để tìm kiếm giá trị k>0 trong dãy
{1,2,15,0,0,0,0,0,37,0,0,0,45,0,54,107,0,0,0}
Nếu biết số lượng số 0 bị xen lẫn chỉ <5% tổng số phần tử thì đâu là cách hiệu quả nhất để tìm kiếm
(1/1 Point)

- [ ] tìm kiếm tuần tự, nhảy bỏ qua phần tử 0
- [x] tìm kiếm nhị phân và bỏ qua phần tử 0
- [ ] lọc lại danh sách và bỏ phần tử 0 trước khi tìm kiếm nhị phân

6.thời gian để tìm phần tử lớn nhất trên cây nhị phân tìm kiếm là
(1/1 Point)

- [x] O(h) với h là chiều cao của cây
- [ ] O(n)

- [ ] O(1)
- [ ] O(n/2)

7.Đây là khẳng định đúng về cây nhị phân tìm kiếm
(0/1 Point)

- [x] Có thể áp dụng tìm kiếm nhị phân trên cấu trúc liên kết
- [x] Có thể thêm xóa phần tử dễ dàng mà không cần dịch
- [x] Luôn duy trì một danh sách các phần tử theo thứ tự (khi duyệt theo thứ tự giữa)
- [ ] Tiết kiệm bộ nhớ hơn so với mảng

8.Đâu là lợi thế của thuật toán tìm kiếm tuần tự
(1/1 Point)

- [x] Danh sách không cần có thứ tự
- [ ] Thời gian tìm kiếm nhanh
- [x] Cài đặt đơn giản
- [x] Có thể áp dụng trên nhiều koaij cấu trúc dữ liệu

9.Những vấn đề nào cần quan tâm khi xử ly bài toán tìm kiếm
(0/1 Point)

- [x] Kích thước danh sách phần tử
- [x] Kiểu giá trị của khóa
- [x] Kiểu bài toán tìm kiếm
- [x] Thời gian tối đa được xử lý trước khi trả về kết quả

10.Làm thế nào để xây dựng được cây nhị phân tìm kiếm với chiều cao thấp nhất từ dãy n phần tử
(1/1 Point)

- [x] Sắp xếp dãy và chọn phần tử giữa làm gốc, sau đó lại xây dựng cây tiếp cho nửa trái và phải
- [ ] Chỉ cần chọn phần tử trung vị của dãy ban đầu làm gốc là đủ
- [ ] Chỉ cần không chọn phần tử lớn nhất hoặc nhỏ nhất trong dãy làm gốc là đủ
- [ ] Không thể xây dựng được cây nhị phân tìm kiếm với chiều cao nhỏ nhất

TUẦN 10

1.Thời gian dể thực hiện thêm nút vào cây nhị phân tìm kiếm chiều cao h với n nút trong trường hợp tồi nhất
(0/1 Point)

- [ ] O(n)

- [ ] O(logn)
- [x] O(h)
- [ ] O(n+h)

2.Nút có giá trị gần nhất với nút bị xóa (trường hợp xóa nút trong) có đặc điểm là
(1/1 Point)

- [x] nút trái nhất trên cây con phải
- [ ] nút lá sâu nhất trên con phải
- [x] nút phải nhất trên cây con trái
- [ ] nút là nông nhất trên con phải

3.Làm thế nào để loại bỏ các khoá nằm ngoài khoảng k1<x<k2 trên cây nhị phân tìm kiếm 1 cách nhanh nhất
(1/1 Point)

- ( ) thực hiện liên tục việc xóa từng nút trên cây nếu khóa nằm ngoài khoảng
- (•) chuyển vây về danh sách có thứ tự, loại các khóa nằm ngoài khoảng và xây lại cây tư danh sách

4.Thời gian để tìm tổ tiên chung gần nhất của 2 nút trên cây nhị phân tìm kiếm là
(1/1 Point)

- [ ] O(logn)
- [ ] O(n^2)
- [x] O(n)
- [ ] O(nlogn)

5.Nút mới thêm vào cây nhị phân tìm kiếm sẽ mọc tại
(1/1 Point)

- [ ] Tại nút trong
- [x] Tại nút lá
- [ ] Tại nút bất kỳ
- [ ] Tại gốc

6.Khi loại bỏ nút trên cây, ta sẽ phải thay nút bị xóa bằng nút con trực tiếp của nó trong trường hợp nào
(1/1 Point)

- [ ] xóa nút lá
- [ ] xóa nút trong có 2 con
- [x] xóa nút trong có 1 con

- [ ] tất cả các trường hợp xóa nút

7.Để trộn 2 cây nhị phân tìm kiếm với khóa không trùng nhau vào thành 1 cây mới, thuật toán nào sẽ cho thời gian thực hiện nhanh hơn

(0/1 Point)

- [x] thêm từng nút của cây nhỏ hơn vào cây lớn hơn
- [ ] thêm từng nút của cây lớn hơn vào cây nhỏ hơn
- [ ] chuyển cây về danh sách có thửu tự, sau đó trộn 2 danh sách có thứ tự với nhau và xây lại cây

8.Cách nào hiệu quả nhất dể tìm và trả về khóa nhỏ thứ k trên cây trong các cách sau?

(1/1 Point)

- [ ] chuyển cây về dãy có thứ tự và trả về phần tử thứ k trong dãy tăng
- [x] duyệt cây theo thứ tự giữa và đếm đến k thì dừng và trả về

9.Khi thêm nút mới vào cây, nếu khóa đã tồn tại rồi chúng ta sẽ phải làm gì?

(1/1 Point)

- [ ] loại bỏ khóa cũ
- [x] không làm gì cả
- [ ] thêm tiếp vào cây con trái của khóa bị trùng
- [ ] thêm tiếp vào cây con phải của khóa bị trùng

10.Làm thế nào để lọc ra các khóa trùng nhau trên 2 cây nhị phân tìm kiếm với thời gian nhanh nhất

(1/1 Point)

- [ ] chuyển cây lớn hơn về danh sách có thứ tự và áp dụng tìm kiếm nhị phân
- [x] tìm tứng nút của cây nhỏ hơn trên cây lớn hơn để tìm các khóa bị trùng
- [ ] tìm tứng nút của cây lớn hơn trên cây nhỏ hơn để tìm các khóa bị trùng