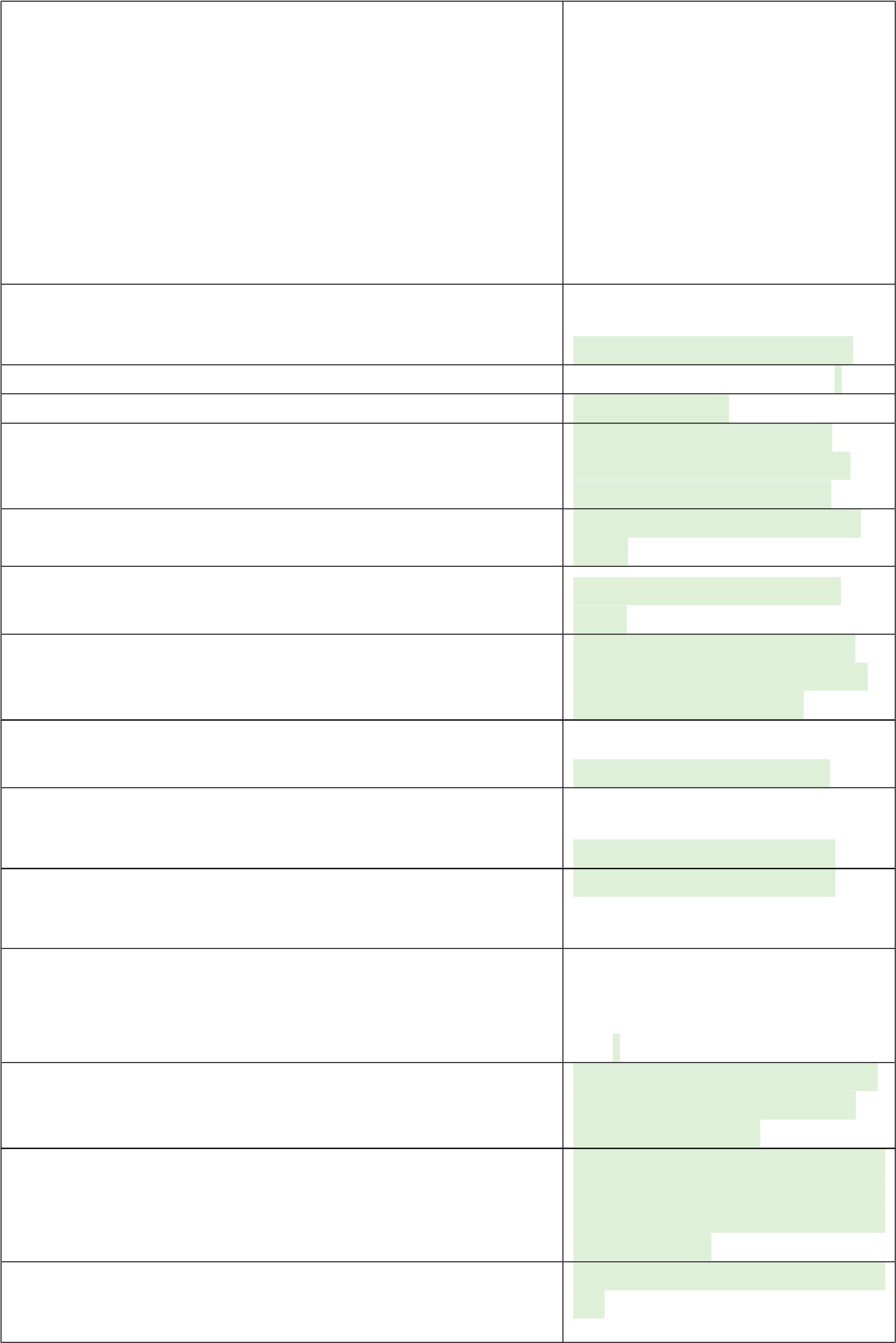
|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT | | | | | | | | | |
| 096.321.3087 | | | | | | | | | |
|  |  |  |  | | | | | | |
|  |
|  | | |  | 8, 6, 14, 16, 11, 33, 31, 46 | | | | | , |
| 36, 30 |  | | | |
|  |
| Bậc của cây có nghĩa là gì? | | |  | | | | | | |
|  | Là bậc lớn nhất của các nút | | | | |  |
| trong cây | | |  | |
| Bậc của nút trong cây có nghĩa là gì? | | |  | Là số nhánh con của nút đó | | | |  | |
|  | | | |
| Các bước thực hiện tìm kiếm nhị phân phần tử x trên dẫy sắp xếp tăng dần được mô tả như sau: Bước 1: Khởi đầu tìm kiếm trên tất cả các phần tử của dãy c left = …………… và right =  ………………  Bước 2: Tính middle = (left + right)/2. So sánh a[middle] với x. Có 3 khả năng: - a[middle] = x => Tìm thấy => Dừng   * a[middle] > x => tiếp tục tìm x trong dãy con mới với right = middle - 1 (tìm trong nửa đầu) * a[middle] < x => tiếp tục tìm x trong dãy con mới   với left = middle + 1 (tìm trong nửa cuối) Bước 3:   * Nếu left <= right => dãy còn phần tử, tiếp tục quay lại bước 2 để tìm kiếm tiếp * Ngược lại => Dãy hiện hành hết phần tử và dừng thuật toán   Giá trị cần điền vào dấu ………….. là bao nhiêu để thuật toán thực hiện đúng | | |  | | | | | | |
|  | 0 và n-1 | |  | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Các bước thực hiện tìm kiếm nhị phân phần tử x trên dẫy sắp xếp tăng dần được mô tả như sau: Bước 1: Khởi đầu tìm kiếm trên tất cả các phần tử của dãy <=> left = 0 và right = n-1  Bước 2: Tính middle = (left + right)/2. So sánh a[middle] với x. Có 3 khả năng: - a[middle] = x => Tìm thấy => Dừng   * a[middle] > x => tiếp tục tìm x trong dãy con mới với right = middle - 1 (tìm trong nửa đầu) * a[middle] < x => tiếp tục tìm x trong dãy con mới với ............................ (tìm trong nửa cuối)   Bước 3:   * Nếu left <= right => dãy còn phần tử, tiếp tục quay lại bước 2 để tìm kiếm tiếp * Ngược lại => Dãy hiện hành hết phần tử và dừng thuật toán   Giá trị cần điền vào dấu ………….. là bao nhiêu để thuật toán thực hiện đúng |  | | | | |
|  | left = middle + 1 | | |  |
| Các bước thực hiện tìm kiếm nhị phân phần tử x trên dẫy sắp xếp tăng dần được mô tả như sau: Bước 1: Khởi đầu tìm kiếm trên tất cả các phần tử của dãy c left = …………… và right =  ………………  Bước 2: Tính middle = (left + right)/2. So sánh a[middle] với x. Có 3 khả năng: - a[middle] = x => Tìm thấy => Dừng   * a[middle] > x => tiếp tục tìm x trong dãy con mới với right = middle - 1 (tìm trong nửa đầu) * a[middle] < x => tiếp tục tìm x trong dãy con mới   với left = middle + 1 (tìm trong nửa cuối) Bước 3:   * Nếu left <= right => dãy còn phần tử, tiếp tục quay lại bước 2 để tìm kiếm tiếp * Ngược lại => Dãy hiện hành hết phần tử và dừng thuật toán   Giá trị cần điền vào dấu ………….. là bao nhiêu để thuật toán thực hiện đúng |  | 0 và n-1 |  | | |
|  |
| Các bước thực hiện tìm kiếm nhị phân phần tử x trên dẫy sắp xếp tăng dần được mô tả như sau: Bước 1: Khởi đầu tìm kiếm trên tất cả các phần tử của dãy <=> left = 0 và right = n-1  Bước 2: Tính middle = (left + right)/2. So sánh a[middle] với x. Có 3 khả năng: - a[middle] = x => Tìm thấy => Dừng  - a[middle] > x => tiếp tục tìm x trong dãy con mới với right = middle - 1 (tìm trong nửa đầu) |  | left = middle + 1 | |  | |
|  | |

* a[middle] < x => tiếp tục tìm x trong dãy con mới với ............................ (tìm trong nửa cuối)

Bước 3:

* Nếu left <= right => dãy còn phần tử, tiếp tục quay lại bước 2 để tìm kiếm tiếp - Ngược lại => Dãy hiện hành hết phần tử và dừng thuật toán

Giá trị cần điền vào dấu ………….. là bao nhiêu để thuật toán thực hiện đúng

Các dạng biểu diễn của biểu thức toán học gồm

Tiền tố, trung tố và hậu tố

Các hàm để cấp phát bộ nhớ là?. malloc(), calloc(), new() Các hàm để giải phóng bộ nhớ là delete(),free().

Danh sách liên kết đơn, danh sách liên kết kép và

Các loại danh sách liên kết gồm: danh sách liên kết vòng

Tìm kiếm tuyến tính và nhị

Các phương pháp tìm kiếm là phân

Dữ liệu (data) và liên kết

Các thành phần của danh sách đơn gồm: (link)

Dữ liệu (infor), liên kết với nút trước (previous) và liên

Các thành phần của danh sách liên kết kép gồm: kết với nút sau (next)

Các thao tác cơ bản trên danh sách gồm thao tác

gì: Tất cả các thao tác trên

+Các thao tác được định nghĩa cho hàng đợi một

cách tổng quát Cả hai đáp án đều đúng

Các thao tác được định nghĩa cho ngăn xếp một Cả hai đáp án đều đúng cách tổng quát

Các trường hợp chèn thêm một phần tử mới vào Chèn thêm vào đầu danh danh sách liên kết đơn gồm: sách, vào cuối danh sách và

vào sau một phần tử q đã

biết

Nút xóa là nút lá, nút xóa có

Các trường hợp có thể xảy ra khi xóa một phần một nhánh con và nút xóa tử khỏi cây NPTK gồm: có hai nhánh con

Các trường hợp thực hiện hủy phần tử khỏi danh Hủy phần tử đầu danh sách, sách liên kết đơn gồm: hủy phần tử đứng sau phần

Các ứng dụng cơ bản của hàng đợi gồm

tử q và hủy phần tử có giá trị xác định k

Tất cả các phương án đều

sai

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Cây là đồ thị vô hướng liên thông |  | | | | | |
|  | Không có chu trình | |  | | |
| Cho biết các nút có bậc bằng 2 trong hình ảnh  sau: | 30, 35, 50, 40 | | | | | |
| Cho biết đây là ý tưởng của thuật toán nào: Xuất phát từ dãy đầu a0, a1, …, ai, xét các phần tử sau đó từ ai+1 đến an xem có phần tử nào nhỏ hơn ai không thì hoán đổi vị trí => Sau mỗi lần luôn được dãy a0, a1, …, ai đã được sắp thứ tự |  | Ý tưởng của thuật toán sắp | | | |  |
| xếp InterchangeSort | | |  |
|  | | |
| Cho biết kết quả của đoạn chương trình sau:  long f5(int n)  {  if (2\*n==2) return 2; else  return 2\*n + f5(n-1);  }  int main()  {  long x = f5(3); printf("%ld", x); getch();  } |  | 12 |  | | | |
|  |
| Cho biết kết quả của đoạn chương trình sau: int F(int a[], int n)  {  if (n==1) return a[0]; else  return a[n-1] + F(a,n-1);  }  int main()  {  int a[] = {2, 3, 4, 5, 6}; printf("%d",F(a,5)); getch();  } |  | 20 |  | | | |
|  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | | | |
| Cho biết kết quả của đoạn chương trình sau: int F(int a[], int n)  {  if (n==1) return a[0]; else  return 1 + F(a,n-1);  }  int main()  {  int a[] = {2, 3, 4, 5, 6}; printf("%d",F(a,5)); getch();  } |  | 6 |  | |
|  |
| Cho biết kết quả của đoạn chương trình sau:  long f3(int n)  {  if (n==1) return 1; else  return n\*n + f3(n-1);  }  int main()  {  long x = f3(3); printf("%ld", x); getch();  } |  | 14 | |  |
|  | |
| Cho biết kết quả của đoạn chương trình sau: int F(int a[], int n)  {  if (n==1) return a[0]; else  return a[n-1] + F(a,n-1);  }  int main()  {  int a[] = {2, 3, 4, 5, 6}; printf("%d",F(a,5)); getch();  } | 20 | | | |

|  |  |
| --- | --- |
| Cho biết kết quả của đoạn chương trình sau: int F(int a[], int n)  {  if (n==1) return a[0]; else  return 1 + F(a,n-1);  }  int main()  {  int a[] = {2, 3, 4, 5, 6}; printf("%d",F(a,5)); getch();  } | 6 |
| Cho biết kết quả của đoạn chương trình sau: int F(int a[], int n)  {  if (n==1) return a[0]; else  return a[n-1] + F(a,n-1);  }  int main()  {  int a[] = {2, 3, 4, 5, 6}; printf("%d",F(a,5)); getch();  } | 20 |
| Cho biết kết quả của đoạn chương trình sau:  long f3(int n)  {  if (n==1) return 1; else  return n\*n + f3(n-1);  }  int main()  {  long x = f3(3); printf("%ld", x); getch();  } | 14 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Cho biết kết quả của đoạn chương trình sau:  long f5(int n)  {  if (2\*n==2) return 2; else  return 2\*n + f5(n-1);  }  int main()  {  long x = f5(3); printf("%ld", x); getch();  } | 12 | | | |
| Cho biết kết quả khi duyệt cây sau bằng phương pháp duyệt LRN (Left-Right-Node)? | 28, 30, 32, 35, 38, 40, 45, 47, 50, 55  40, 35, 30, 28, 32, 38, 50,  45, 47, 55 | | | |
| Cho biết kết quả sau khi thực hiện đoạn chương trình sau: int main()  {  int a[20], n,i,k; k = a[0]; for(i=0; i<n; i++)  if (a[i] > k) k = a[i];  } |  | k có giá trị lớn nhất |  | |
|  |
|  | |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Cho biết kết quả sau khi thực hiện đoạn chương trình sau: int main()  {  int a[20], n,i,k; k = a[n-1]; for(i=n-2; i>=0; i--)  if (a[i] < k) k = a[i];  } | k có giá trị nhỏ nhất | | | |
| Cho biết kết quả sau khi thực hiện đoạn chương trình sau: int main()  {  int a[20], n,i,k; k = a[0]; for(i=0; i<n; i++)  if (a[i] > k) k = a[i];  } | k có giá trị lớn nhất | | | |
| Cho biết kết xuất của đoạn chương trình sau:  long F(int n)  {  if ((2\*n+1) ==1) return 1; else  return (2\*n+1)+F(n-1);  }  void main()  {  long x=F(3); printf("%ld", x);  } |  | 16 |  | |
|  |
| Cho các bước mô tả thuật toán như sau:  Nếu danh sách rỗng:  DQ.Head = new\_element; DQ.Tail = DQ.Head; Ngược lại (d/s khác rỗng):  new\_element -> next = DQ.Head;  DQ.Head -> pre = new\_element;  DQ.Head = new\_element;  Đây là mô tả của thuật toán chèn một phần tử vào danh sách liên kết đôi với vị trí chèn là? |  | Chèn vào đầu danh sách | |  |
|  | |
| Cho các phần tử 5, 10, 3, 42 lần lượt được bổ sung vào hàng đợi (Queue). Phần tử nào được lấy ra đầu tiên | 5 | | | |
| Cho mảng a gồm các phần tử: 8, 3, 7, 6, 4, 2.  Cho biết kết quả ở bước thứ 3 khi áp dụng thuật toán sắp xếp Selection tăng dần trên mảng các phần tử trên.  Chọn một câu trả lời: | 2.3.4.6.7.8 | | | |
| Cho mảng a gồm các phần tử: 8, 3, 7, 6, 4, 2.  Cho biết kết quả ở bước thứ 3 khi áp dụng thuật toán sắp xếp Selection tăng dần trên mảng các phần tử trên.  Chọn một câu trả lời: | 3.6.7.8.4.2 | | | |
| Cho đoạn mã sau:  struct CB{ int mcb; char hoten[20]; char ns[12]; float hsl,pc,tt;};struct Node{ CB info; struct Node \*next;};  struct List{ Node \*head, \*tail;};  Khai báo Cấu trúc dữ liệu trên là khai báo CTDL dạng gì? | Danh sách liên kết đôi | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Cho các phần tử 5, 10, 3, 42 lần lượt được bổ sung vào hàng đợi (Queue). Phần tử nào được lấy ra cuối cùng |  | | | | |
|  | 42 |  | | |
| Cho các phần tử 5, 10, 3, 42 lần lượt được bổ sung vào hàng đợi (Queue). Phần tử nào được lấy ra đầu tiên | 5 | | | | |
| Cho các phần tử 5, 10, 3, 42 lần lượt được bổ sung vào ngăn xếp (Stack). Phần tử nào được lấy ra đầu tiên |  | | | | |
|  | 42 |  | | |
| Cho các phần tử 5, 10, 3, 42 lần lượt được bổ sung vào ngăn xếp (Stack). Phần tử nào được lấy ra cuối cùng | 5 | | | | |
| Cho các phần tử 5, 10, 3, 42 lần lượt được bổ sung vào ngăn xếp (Stack). Phần tử nào được lấy ra đầu tiên |  | 42 |  | | |
|  |
| Cho các phần tử 5, 10, 3, 42 lần lượt được bổ sung vào ngăn xếp (Stack). Phần tử nào được lấy ra cuối cùng |  | 10 |  | | |
| 42 |
| Cho các phần tử sau: 31, 19, 36, 20, 41, 17, 33, 32. Tạo cây NPTK từ các phần tử trên. Hãy cho biết sau khi xóa phần tử 33 trên cây sau đó áp dụng phương pháp duyệt RNL thì kết quả thu được thứ tự các phần tử là như thế nào? |  | | | | |
|  | 41, 36, 32, 31, 20, 19, 17 | |  | |
| Cho các phần tử sau: 31, 19, 36, 20, 41, 17, 33, 32. Tạo cây NPTK từ các phần tử trên. Hãy cho biết sau khi xóa phần tử 33 trên cây sau đó áp dụng phương pháp duyệt NRL thì kết quả thu được thứ tự các phần tử là như thế nào? |  | | | | |
|  | 31, 36, 41, 32, 19, 20, 17 | | |  |
| Cho các phần tử sau: 31, 19, 36, 20, 41, 17, 33, 32. Tạo cây NPTK từ các phần tử trên. Hãy cho biết sau khi xóa phần tử 33 trên cây sau đó áp dụng phương pháp duyệt NLR thì kết quả thu được thứ tự các phần tử là như thế nào? |  | | | | |
|  | 31, 19, 17, 20, 36, 32, 41 | |  | |
| Cho các phần tử sau: 31, 19, 36, 20, 41, 17, 33, 32. Tạo cây NPTK từ các phần tử trên. Hãy cho biết sau khi xóa phần tử 33 trên cây sau đó áp dụng phương pháp duyệt LNR thì kết quả thu được thứ tự các phần tử là như thế nào? |  | | | | |
|  | 17, 19, 20, 31, 32, 36, 41 | |  | |
| Cho các phần tử sau: 31, 19, 36, 20, 41, 17, 33, 32. Tạo cây NPTK từ các phần tử trên. Hãy cho biết sau khi xóa phần tử 33 trên cây sau đó áp dụng phương pháp duyệt RLN thì kết quả thu được thứ tự các phần tử là như thế nào |  | | | | |
|  | 41, 32, 36, 20, 17, 19, 31 | |  | |
| Cho các phần tử sau: 31, 19, 36, 20, 41, 17, 33, 32. Tạo cây NPTK từ các phần tử trên. Hãy cho biết sau khi xóa phần tử 33 trên cây sau đó áp dụng phương pháp duyệt NLR thì kết quả thu |  | 31, 19, 17, 20, 36, 32, 41 | |  | |
|  | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| được thứ tự các phần tử là như thế nào? |  | | | | | |
| Cho các phần tử sau: 31, 19, 36, 20, 41, 17, 33, 32. Tạo cây NPTK từ các phần tử trên. Hãy cho biết sau khi xóa phần tử 33 trên cây sau đó áp dụng phương pháp duyệt LNR thì kết quả thu được thứ tự các phần tử là như thế nào? |  | 17, 19, 20, 31, 32, 36, 41 | |  | | |
|  | |
| Cho các phần tử sau: 31, 19, 36, 20, 41, 17, 33, 32. Tạo cây NPTK từ các phần tử trên. Hãy cho biết sau khi xóa phần tử 33 trên cây sau đó áp dụng phương pháp duyệt NRL thì kết quả thu được thứ tự các phần tử là như thế nào? |  | 31, 19, 36, 20, 41, 17, 32 | |  | | |
| 41, 32, 36, 20, 17, 19, 31 | |
|  | |
| Cho các phần tử sau: 31, 19, 36, 20, 41, 17, 33, 32. Tạo cây NPTK từ các phần tử trên. Hãy cho biết sau khi xóa phần tử 33 trên cây sau đó áp dụng phương pháp duyệt RNL thì kết quả thu được thứ tự các phần tử là như thế nào? |  | 31, 19, 36, 20, 41, 17, 32 | |  | | |
| 41, 32, 36, 20, 17, 19, 31 | |
|  | |
| Cho cây NPTK, Cho biết kết quả duyệt cây theo thứ tự LNR là: |  | | | | | |
|  | 6, 8, 11, 14, 16, 30, 31, 33, | | |  | |
| 36, 46 |  | |
| Cho cây NPTK, Cho biết kết quả duyệt cây theo thứ tự LNR là: | 6, 8, 11, 14, 16, 30, 31, 33,  36, 46 | | | | | |
| Cho cây NPTK, Cho biết kết quả duyệt cây theo thứ tự LRN là: |  | | | | | |
|  | 8, 6, 14, 16, 11, 33, 31, 46, | | | |  |
| 36, 30 |  | | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Cho cây NPTK, Cho biết kết quả duyệt cây theo thứ tự LRN là: |  | 8, 6, 14, 16, 11, 33, 31, 46, | | | | |  |
| 36, 30 |  | | | |
|  |
| Cho cây NPTK, Cho biết kết quả duyệt cây theo thứ tự RNL là: |  | | | | | | |
|  | 46, 36, 33, 31, 30, 16, 14, | | | |  | |
| 11, 8, 6 | |  | |
| Cho cây NPTK, Cho biết kết quả duyệt cây theo thứ tự RNL là: |  | 46, 36, 33, 31, 30, 16, 14, | | | | |  |
| 11, 8, 6 | |  | | |
|  | |
| Cho cây NPTK, chọn biểu thức tương ứng với cây: | (3+4)\*((8-2)\*6) | | | | | | |
| Cho cây NPTK, chọn biểu thức tương ứng với cây: | (3+4)\*((8-2)\*6) | | | | | | |
| Cho dãy 10, 5, 7, 3, 9, 2, 15, 1. Cho biết kết quả sau lần duyệt thứ nhất của thuật toán sắp xếp tăng dần bằng QuickSort |  | | | | | | |
|  | 1, 2, 3,7,9, 5, 15, 10 | | |  | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Cho dãy 10, 5, 7, 3, 9, 2, 15, 1. Cho biết kết quả sau lần duyệt thứ nhất của thuật toán sắp xếp tăng dần bằng QuickSort |  | 1, 2, 3,7,9, 5, 15, 10 | |  | |
|  | |
| Cho dãy 10, 5, 7, 3, 9, 2, 15, 1. Dùng thuật toán sắp xếp tăng dần bằng QuickSort, cho biết ở lần duyệt thứ nhất giá trị của x, L và R là gì? |  | | | | |
|  | L=0; R=7; x=3; |  | | |
| Cho dãy 10, 5, 7, 3, 9, 2, 15, 1. Dùng thuật toán sắp xếp tăng dần bằng QuickSort, cho biết ở lần duyệt thứ nhất giá trị của x, L và R là gì? |  | L=0; R=7; x=3; |  | | |
|  |
| Cho dãy sau: 23, 78, 45, 8, 32, 56. Dùng phương pháp sắp xếp chọn trực tiếp (Selection Sort) để sắp xếp tăng dần, sau 2 lần lặp thì kết quả của dãy là thế nào? | 8, 23, 45, 78, 32, 56 | | | | |
| Cho dãy sau: 23, 78, 45, 8, 32, 56. Dùng phương pháp sắp xếp chọn trực tiếp (Selection Sort) để sắp xếp tăng dần, sau 5 lần lặp thì kết quả của dãy là thế nào? |  | | | | |
|  | 8, 23, 32, 45, 56, 78 | |  | |
| Cho dãy sau: 23, 78, 45, 8, 32, 56. Dùng phương pháp sắp xếp chọn trực tiếp (Selection Sort) để sắp xếp tăng dần, sau 3 lần lặp thì kết quả của dãy là thế nào? |  | | | | |
|  | 8, 23, 32, 78, 45, 56 | |  | |
| Cho dãy sau: 23, 78, 45, 8, 32, 56. Dùng phương pháp sắp xếp chọn trực tiếp (Selection Sort) để sắp xếp tăng dần, sau 5 lần lặp thì kết quả của dãy là thế nào? |  | 8, 23, 32, 45, 56, 78 | |  | |
|  | |
| Cho dãy sau: 23, 78, 45, 8, 32, 56. Dùng phương pháp sắp xếp chọn trực tiếp (Selection Sort) để sắp xếp tăng dần, sau 3 lần lặp thì kết quả của dãy là thế nào? |  | 8, 23, 32, 78, 45, 56 | |  | |
|  | |
| Cho dãy sau: 42, 23, 74, 11, 65, 58. Dùng phương pháp sắp xếp chèn trực tiếp (Insertion Sort) để sắp xếp tăng dần, sau 2 lần lặp kết quả của dãy là thế nào? | 23, 42, 74, 11, 65, 58 | | | | |
| Cho dãy sau: 42, 23, 74, 11, 65, 58. Dùng phương pháp sắp xếp chèn trực tiếp (Insertion Sort) để sắp xếp tăng dần, sau 3 lần lặp kết quả của dãy là thế nào? |  | | | | |
|  | 11, 23, 42, 74, 65, 58 | | |  |
| Cho dãy sau: 42, 23, 74, 11, 65, 58. Dùng phương pháp sắp xếp chèn trực tiếp (Insertion Sort) để sắp xếp tăng dần, sau 5 lần lặp kết quả của dãy là thế nào? |  | | | | |
|  | 11, 23, 42, 58, 65, 74 | | |  |
| Cho dãy sau: 42, 23, 74, 11, 65, 58. Dùng phương pháp sắp xếp chèn trực tiếp (Insertion Sort) để sắp xếp tăng dần, sau 2 lần lặp kết quả của dãy là thế nào? |  | 23, 42, 74, 11, 65, 58 | | |  |
|  | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Cho dãy sau: 42, 23, 74, 11, 65, 58. Dùng phương pháp sắp xếp chèn trực tiếp (Insertion Sort) để sắp xếp tăng dần, sau 3 lần lặp kết quả của dãy là thế nào? |  | 11, 23, 42, 74, 65, 58 |  | |
|  |
| Cho dãy sau: 42, 23, 74, 11, 65, 58. Dùng phương pháp sắp xếp chọn trực tiếp (Selection Sort) để sắp xếp giảm dần, sau lần lặp thứ tư kết quả của dãy là thế nào? |  | 74, 65, 58, 42, 23, 11 |  | |
|  |
| Cho dãy sau: 42, 23, 74, 11, 65, 58. Dùng phương pháp sắp xếp đổi chỗ trực tiếp  (Interchange Sort) để sắp xếp tăng dần, sau 3 lần lặp kết quả của dãy là thế nào? |  | | | |
|  | 11, 23, 42, 74, 65, 58 |  | |
| Cho dãy sau: 42, 23, 74, 11, 65, 58. Dùng phương pháp sắp xếp đổi chỗ trực tiếp  (Interchange Sort) để sắp xếp tăng dần, sau 4 lần lặp kết quả của dãy là thế nào? |  | | | |
|  | 11, 23, 42, 58, 74, 65 |  | |
| Cho dãy sau: 42, 23, 74, 11, 65, 58. Dùng phương pháp sắp xếp đổi chỗ trực tiếp  (Interchange Sort) để sắp xếp tăng dần, sau 3 lần lặp kết quả của dãy là thế nào? |  | 11, 23, 42, 74, 65, 58 |  | |
|  |
| Cho dãy sau: 42, 23, 74, 11, 65, 58. Dùng phương pháp sắp xếp đổi chỗ trực tiếp  (Interchange Sort) để sắp xếp tăng dần, sau 4 lần lặp kết quả của dãy là thế nào? |  | 11, 23, 42, 58, 74, 65 |  | |
|  |
| Cho dãy sau: 42, 23, 74, 11, 65, 58. Dùng phương pháp sắp xếp nổi bọt (Bubble Sort) để sắp xếp giảm dần, sau lần lặp thứ ba kết quả của dãy là thế nào? |  | 74, 65, 58, 42, 23, 11 |  | |
|  |
| Cho dãy sau: 42, 23, 74, 11, 65, 58. Dùng phương pháp sắp xếp nổi bọt (Bubble Sort) để sắp xếp tăng dần, sau 1 lần lặp kết quả của dãy là thế nào? |  | | | |
|  | 11, 42, 23, 74, 58, 65 |  | |
| Cho dãy sau: 42, 23, 74, 11, 65, 58. Dùng phương pháp sắp xếp nổi bọt (Bubble Sort) để sắp xếp giảm dần, sau lần lặp thứ ba kết quả của dãy là thế nào? |  | | | |
|  | 74, 65, 58, 42, 23, 11 |  | |
| Cho dãy sau: 42, 23, 74, 11, 65, 58. Dùng phương pháp sắp xếp nổi bọt (Bubble Sort) để sắp xếp tăng dần, sau 4 lần lặp kết quả của dãy là thế nào? |  | | | |
|  | 42, 23, 74, 11, 65, 58 |  | |
| 11, 23, 58, 42, 65, 74 | |  |
| Cho dãy sau: 42, 23, 74, 11, 65, 58. Dùng phương pháp sắp xếp nổi bọt (Bubble Sort) để sắp xếp tăng dần, sau 1 lần lặp kết quả của dãy là thế nào? |  | 11, 42, 23, 74, 58, 65 |  | |
|  |
| Cho dãy sau: 42, 23, 74, 11, 65, 58. Dùng phương pháp sắp xếp nổi bọt (Bubble Sort) để sắp xếp tăng dần, sau 4 lần lặp kết quả của dãy |  | 11, 23, 42, 58, 65, 74 |  | |
|  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| là thế nào? |  | | | | | | |
| Cho dãy sau: 42, 23, 74, 11, 65, 58. Dùng phương pháp sắp xếp phân hoạch (Quick Sort), điểm chốt a[middle] ban đầu là: | a[middle] = 74 | | | | | | |
| Cho dãy sau: 42, 23, 74, 11, 65, 58. Dùng phương pháp sắp xếp phân hoạch (Quick Sort), điểm chốt a[middle] ban đầu là: |  | a[middle] = 74 | |  | | | |
|  | |
| Cho dãy số sau: 30, 18, 35, 17, 40, 16, 32, 31, 43, 19. Cho biết kết quả khi duyệt cây được tạo lần lượt từ các phần tử trên bằng phương pháp duyệt NLR (Node Left Right): |  | | | | | | |
|  | 30, 18, 17, 16, 19, 35, 32 | | | , | | |
| 31, 40, 43 |  | |
| Cho dãy số sau: 30, 18, 35, 17, 40, 16, 32, 31, 43, 19. Cho biết kết quả khi duyệt cây được tạo lần lượt từ các phần tử trên bằng phương pháp duyệt LRN (Left Right Node ): |  | | | | | | |
|  | 16, 17, 19, 18, 31, 32, 43, | | |  | | |
| 40, 35, 30 |  | |
| Cho dãy số sau: 30, 18, 35, 17, 40, 16, 32, 31,  43, 19. Cho biết kết quả khi duyệt cây được tạo lần lượt từ các phần tử trên bằng phương pháp duyệt RNL(Right Node Left): |  | | | | | | |
|  | 43, 40, 35, 32, 31, 30, 19, | | |  | | |
| 18, 17, 16 |  | |
| Cho dãy số sau: 30, 18, 35, 17, 40, 16, 32, 31,  43, 19. Cho biết kết quả khi duyệt cây được tạo lần lượt từ các phần tử trên bằng phương pháp duyệt LRN (Left Right Node ): |  | 16, 17, 19, 18, 31, 32, 43, | | | | |  |
| 40, 35, 30 |  | | | |
|  |
| Cho dãy số sau: 30, 18, 35, 17, 40, 16, 32, 31, 43, 19. Cho biết kết quả khi duyệt cây được tạo lần lượt từ các phần tử trên bằng phương pháp duyệt NLR (Node Left Right): |  | 30, 18, 17, 16, 19, 35, 32, | | | | |  |
| 31, 40, 43 |  | | | |
|  |
| Cho đồ thị G = <V, > dưới dạng ma trận trọng số. Hãy cho biết đâu là tập cạnh của cây khung nhỏ nhất được xây dựng theo thuật toán Kruskal |  | | | | | | |
|  | T={ (2, 3), (1, 3), (4, 5), (4, | | | |  | |
| 6), (3, 5) } |  | | |
| Cho đồ thị G = <V, > dưới dạng ma trận trọng số. Hãy cho biết đâu là tập cạnh của cây khung nhỏ nhất được xây dựng theo thuật toán Kruskal |  | T={ (2, 3), (1, 3), (4, 5), (4, | | | | |  |
| 6), (3, 5) } |  | | | |
|  |

|  |  |
| --- | --- |
| Cho đồ thị sau:  Cho biết ma  trận kề biểu diễn đồ thị trên là gì? | s ai |
| Cho đồ thị sau:  Cho biết ma  trận trọng số biểu diễn đồ thị trên là gì? |  |
| Cho đồ thị trọng số G = <V, > như hình v . Hãy cho biết đâu là tập cạnh của cây bao trùm ngắn nhất được xây dựng theo thuật toán Prim. Giả sử bắt đầu từ đỉnh 1 | T = { (1, 4), (4, 3), (1, 2), (4, 5), (2, 6), (6,7) } |
| Cho đồ thị trọng số G = <V, > như hình v . Hãy cho biết đâu là tập cạnh của cây bao trùm ngắn nhất được xây dựng theo thuật toán Prim. Giả sử bắt đầu từ đỉnh 1 | T = { (1, 4), (4, 3), (1, 2), (4, 5), (2, 6), (6,7) } |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | | | |
| Cho đồ thị vô hướng như hình v . Đỉnh nào dưới đây là đỉnh cô lập của đồ thị | Đỉnh b | | | |
| Cho đồ thị vô hướng như hình v . Đỉnh nào dưới đây là đỉnh cô lập của đồ thị |  | Đỉnh f |  | |
|  |
| Cho đồ thị vô hướng như hình v . Đỉnh nào dưới đây là đỉnh treo của đồ thị: |  | | | |
|  | Đỉnh f |  | |
| Cho đồ thị vô hướng như hình v . Đỉnh nào dưới đây là đỉnh treo của đồ thị: |  | Đỉnh f | |  |
|  | |
| Cho đồ thị vô hướng như hình v . Hãy cho biết ma trận kề nào là biểu diễn đúng của đồ thị |  | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Cho đồ thị vô hướng như hình v . Hãy cho biết ma trận kề nào là biểu diễn đúng của đồ thị |  | | | |
| Cho đoạn chương trình như sau: void AddAfter(DLIST &DQ, DNode \*q, DNode  \*new\_element)  {  DNode \*p = q -> next;  if (q != NULL)  {  new\_element -> next = p; new\_element -> pre = q; q -> next = new\_element;  if (p != NULL)  …[1]…  if (q == DQ.Tail)  DQ.Tail = new\_element;  }  else  AddFirst( DQ, new\_element);  }  Đoạn lệnh nào được điền vào [1] cho đúng?  Chọn một câu trả lời: |  | | | |
|  | p -> pre = new\_element; | |  |
| Cho đoạn chương trình như sau: void RemoveHead( DLIST &DQ )  {  DNode\*p;  if ( DQ.Head != NULL)  {  p = DQ.Head;  DQ.Head = DQ.Head -> next;  (...1...) free(p);  if ( DQ.Head == NULL)DQ.Tail = NULL;  }  }  Đoạn lệnh được đưa vào (1) là? |  | | | |
|  | DQ.Head -> pre = NULL; | |  |
| Cho đoạn chương trình như sau: void AddAfter(DLIST &DQ, DNode \*q, DNode  \*new\_element)  {  DNode \*p = q -> next; if (q != NULL) |  | DQ.Tail = new\_element; |  | |
|  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| {  new\_element -> next = p; new\_element -> pre = q; q -> next = new\_element; if (p != NULL) p -> pre = new\_element;  if (q == DQ.Tail)  …[1]…  }  else  AddFirst( DQ, new\_element);  }  Đoạn lệnh nào được điền vào [1] cho đúng? |  | | | |
| Cho đoạn chương trình sau:  void RemoveTail ( DLIST &DQ )  {  DNode \*p;  if ( DQ.Tail != NULL)  {  p = DQ.Tail;  ..(1).. free(p);  if ( DQ.Head == NULL) DQ.Tail = NULL; else DQ.Head ->pre = NULL;  }  } |  | | | |
|  | DQ.Tail = DQ.Tail -> pre; | |  |
| DQ.Tail -> next = NULL; |  |
| Cho đoạn chương trình:  void QuickSort( int a[ ], int L , int R )  { int i,j,x;  x=……..;  i = L; j = R; do  {  while ( a[i] < x ) i++; while ( a[j] > x ) j--;  if ( i <= j )  {  Hoanvi (a[i], a[j]);  i++; j--;  }  } while(i<j); if (L<j) QuickSort(a,L,j);  if (i<R) QuickSort(a,i,R);  }  Điền giá trị nào vào đoạn …. cho đúng | a[(L+R)/2] | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | | | |
| Cho đoạn chương trình:  void QuickSort( int a[ ], int L , int R )  { int i,j,x;  x= a[(L+R)/2]; i =…;  j = ...;  do  {  while ( a[i] < x ) i++; while ( a[j] > x ) j--;  if ( i <= j )  {  Hoanvi (a[i], a[j]);  i++; j--;  }  } while(i<j); if (L<j) QuickSort(a,L,j);  if (i<R) QuickSort(a,i,R);  }  Điền giá trị nào vào đoạn …. cho đúng |  | | | |
|  | i=L; j=R; |  | |
| Cho đoạn chương trình:  void QuickSort( int a[ ], int L , int R )  { int i,j,x;  x= a[(L+R)/2];  i = L; j = R; do  {  while ( a[i] < x ) i++; while ( a[j] > x ) j--;  if ( i <= j )  {  Hoanvi (a[i], a[j]);  i++; j--;  }  } while(i<j); if (L<j) …. if (i<R) ….  }  Điền giá trị nào vào đoạn …. cho đúng |  | | | |
|  | QuickSort(a,L,j); | |  |
| QuickSort(a,i,R); | |

|  |  |  |  |
| --- | --- | --- | --- |
| Cho đoạn chương trình:  void QuickSort( int a[ ], int L , int R )  { int i,j,x;  x= a[(L+R)/2]; i =…;  j = ...;  do  {  while ( a[i] < x ) i++; while ( a[j] > x ) j--;  if ( i <= j )  {  Hoanvi (a[i], a[j]);  i++; j--;  }  } while(i<j);  if (L<j) QuickSort(a,L,j); if (i<R) QuickSort(a,i,R);  }  Điền giá trị nào vào đoạn …. cho đúng |  | i=L; j=R; |  |
|  |
| Cho đoạn chương trình:  void QuickSort( int a[ ], int L , int R )  { int i,j,x;  x=……..;  i = L; j = R; do  {  while ( a[i] < x ) i++; while ( a[j] > x ) j--;  if ( i <= j )  {  Hoanvi (a[i], a[j]);  i++; j--;  }  } while(i<j); if (L<j) QuickSort(a,L,j);  if (i<R) QuickSort(a,i,R);  }  Điền giá trị nào vào đoạn …. cho đúng | a[(L+R)/2] | | |
| Cho đoạn mã cài đặt phương pháp duyệt NLR:  void NLR( Tree Root )  {  if ( root != NULL ) | NLR ( Root -> Right ); | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| {  < Xử lý Root >; NLR ( Root -> Left ); NLR(Root->Left) ; [1] ……….  }  }  Đoạn mã điền vào phần trống ở dòng số [1] |  | | | | |
| Cho đoạn mã cài đặt phương pháp duyệt NLR:  void NLR( Tree Root )  {  if ( root != NULL )  {  < Xử lý Root >; NLR ( Root -> Left ); NLR(Root->Left) ; [1] ……….  }  }  Đoạn mã điền vào phần trống ở dòng số [1] |  | NLR ( Root -> Right ); | | |  |
|  | | |
| Cho đoạn mã sau  stack <int> s; for (int i = 1; i <= 5; i++) s.push(i); while (!s.empty()) { cout << s.top() << endl; s.pop(); } Kết quả in lên màn hình là gì? |  | | | | |
|  | 5, 4, 3, 2, 1 | |  | |
| Cho đoạn mã sau  stack <int> s; for (int i = 1; i <= 4; i++) s.push(i);  Phần tử được lấy ra đầu tiên của Stack là gì? | 4 | | | | |
| Cho đoạn mã sau  stack <int> s; for (int i = 1; i <= 5; i++) s.push(i);  Phần tử được lấy ra cuối cùng của Stack là gì? | 1 | | | | |
| Cho đoạn mã sau  stack <int> s; for (int i = 1; i <= 5; i++) s.push(i);  s.pop();  Kết quả các phần tử của Stack sau khi thực hiện các đoạn mã trên là gì? |  | | | | |
|  | 1, 2, 3, 4 |  | | |
| Cho đoạn mã sau  stack <int> s; for (int i = 1; i <= 4; i++) s.push(i); | 4 | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Phần tử được lấy ra đầu tiên của Stack là gì? |  | | | | |
| Cho đoạn mã sau  stack <int> s; for (int i = 1; i <= 5; i++) s.push(i); while (!s.empty()) { cout << s.top() << endl; s.pop(); } Kết quả in lên màn hình là gì? |  | 5, 4, 3, 2, 1 | | |  |
|  | | |
| Cho đoạn mã sau  stack <int> s; for (int i = 1; i <= 5; i++) s.push(i);  s.pop();  Kết quả các phần tử của Stack sau khi thực hiện các đoạn mã trên là gì? |  | 1, 2, 3, 4 |  | | |
|  |
| Cho đoạn mã sau  stack <int> s; for (int i = 1; i <= 5; i++) s.push(i);  Phần tử được lấy ra cuối cùng của Stack là gì? | 1 | | | | |
| Cho đoạn mã sau, cho biết đoạn mã biểu diễn thuật toán gì? Bước 1: S = 1, i = 1;  Bước 2: Nếu i<n thì s = s\*i, qua bước 3;  Ngược lại qua bước 4;  Bước 3: i = i + 1;  Quay lại bước 2;  Bước 4: Xuất S ra màn hình |  | Tính (n-1)! | |  | |
|  | |
| Cho đoạn mã sau, cho biết đoạn mã biểu diễn thuật toán gì? Bước 1: S = 1, i = 1;  Bước 2: Nếu i<n thì s = s\*i, qua bước 3;  Ngược lại qua bước 4;  Bước 3: i = i + 1;  Quay lại bước 2;  Bước 4: Xuất S ra màn hình | Tính (n-1)! | | | | |
| Cho đoạn mã sau, cho biết kết quả của x?  Queue Q;  InitQueue(Q);  Put(Q, “Green”);  Put(Q, “Red”);  Put(Q, “Yellow”);  Get(Q,x); | Green | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Cho đoạn mã sau, cho biết kết quả của x?  Queue Q;  InitQueue(Q);  Put(Q, “Green”);  Put(Q, “Red”);  Put(Q, “Yellow”);  Get(Q,x);  Get(Q,x); |  | | | | |
|  | Red | |  | |
| Cho đoạn mã sau, cho biết kết quả của x?  Stack S;  InitStack(S);  Push(S, “Green”);  Push(S, “Red”); Push(S, “Yellow”);  Pop(S,x); |  | | | | |
|  | Yellow | | |  |
| Cho đoạn mã sau, cho biết kết quả của x?  Stack S;  InitStack(S);  Push(S, “Green”);  Push(S, “Red”);  Push(S, “Yellow”);  Pop(S,x);  Pop(S, x); |  | | | | |
|  | Red |  | | |
| Cho đoạn mã sau, cho biết kết quả của x?  Queue Q;  InitQueue(Q);  Put(Q, “Green”);  Put(Q, “Red”);  Put(Q, “Yellow”);  Get(Q,x); |  | Green | | |  |
|  | | |
| Cho đoạn mã sau, cho biết kết quả của x?  Queue Q;  InitQueue(Q);  Put(Q, “Green”);  Put(Q, “Red”);  Put(Q, “Yellow”);  Get(Q,x);  Get(Q,x); |  | Red |  | | |
|  |
| Cho đoạn mã sau, cho biết kết quả của x?  Stack S;  InitStack(S);  Push(S, “Green”);  Push(S, “Red”);  Push(S, “Yellow”); |  | Yellow | | |  |
|  | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Pop(S,x); |  | | | | | |
| Cho đoạn mã sau, cho biết kết quả của x?  Stack S;  InitStack(S);  Push(S, “Green”);  Push(S, “Red”);  Push(S, “Yellow”);  Pop(S,x);  Pop(S, x); | Red | | | | | |
| Cho đoạn mô tả sau:  Bước 1: Khởi đầu tìm kiếm trên tất cả các phần tử của dãy  (left = 0 và right = n - 1)  Bước 2: Tính middle = (left + right)/2. So sánh a[middle] với x. Có 3 khả năng: a[middle] = x thì thông báo Tìm thấy => Dừng  a[middle] > x thì right = middle - 1 a[middle] < x thì left = middle + 1 Bước 3:  Nếu left <= right và quay lại bước 2 để tìm kiếm tiếp  Ngược lại thông báo không tìm thấy và dừng thuật toán |  | Mô tả thuật toán tìm kiếm | | | |  |
| nhị phân | |  | |
|  | |
| Cho đoạn mô tả sau:  Bước 1: Khởi đầu tìm kiếm trên tất cả các phần tử của dãy  (left = 0 và right = n - 1)  Bước 2: Tính middle = (left + right)/2. So sánh a[middle] với x. Có 3 khả năng: a[middle] = x thì thông báo Tìm thấy => Dừng  a[middle] > x thì right = middle - 1 a[middle] < x thì left = middle + 1 Bước 3:  Nếu left <= right và quay lại bước 2 để tìm kiếm tiếp  Ngược lại thông báo không tìm thấy và dừng thuật toán | Mô tả thuật toán tìm kiếm nhị phân | | | | | |
| Cho G =<V, > là đồ thị vô hướng liên thông n đỉnh. T =<V, H> được gọi là cây khung của đồ thị nếu: |  | | | | | |
|  | T liên thông và có đúng n-1 | | |  | |
| cạnh. |  | |
| Cho G =<V, > là đồ thị vô hướng liên thông n đỉnh. T =<V, H> được gọi là cây khung của đồ thị nếu: |  | T liên thông và có đúng n-1 | | | |  |
| cạnh. |  | | |
|  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Cho hàm tìm kiếm tuyến tính trong mảng 1 chiều có n phần tử  int Search( int a[], int n, int x)  { int i;  for(i=0; i<n; i++) if(a[i] == x) return i;  return(-1);  }  Chọn phát biểu đúng nhất trong các phát biểu sau |  | Hàm trả về vị trí phần tử | | | |  |
| đầu tiên có giá trị bằng x, | | | |
| ngược lại trả về -1 |  | | |
|  |
| Cho hàm tìm kiếm tuyến tính trong mảng 1 chiều có n phần tử  int Search( int a[], int n, int x)  { int i;  for(i=0; i<n; i++) if(a[i] == x) return i;  return(-1);  }  Chọn phát biểu đúng nhất trong các phát biểu sau |  | | | | | |
|  | Hàm trả về vị trí phần tử | |  | | |
| đầu tiên có giá trị bằng x, | | |  | |
| ngược lại trả về -1 |  | |
| Cho mảng a có N (N>=2) phần từ, x là một biến, xét đoạn mã sau cho biết đoạn mã biểu diễn thuật toán gì?  Bước 1: Khởi gán i = 0, s = 0, qua bước 2;  Bước 2: Nếu a[i] == x thì s++; qua bước 3  Bước 3: i = i + 1;  Nếu i == n: hết mảng. Dừng, in s ra màn hình  Ngược lại: Lặp lại bước 2 | Đếm số phần tử có giá trị bằng x trong mảng | | | | | |
| Cho mảng a có N (N>=2) phần từ, x là một biến, xét đoạn mã sau cho biết đoạn mã biểu diễn thuật toán gì?  Bước 1: Khởi gán i = 0, s = 0, qua bước 2;  Bước 2: Nếu a[i] == x thì s++; qua bước 3  Bước 3: i = i + 1;  Nếu i == n: hết mảng. Dừng, in s ra màn hình  Ngược lại: Lặp lại bước 2 |  | Đếm số phần tử có giá trị | | | |  |
| bằng x trong mảng |  | | |
|  |

|  |  |  |  |
| --- | --- | --- | --- |
| Cho mảng a gồm các phẩn tử có giá trị như sau: 74326  Số lần hoán vị 2 phần tử khác nhau khi áp dụng thuật toán chọn trực tiếp để sắp xếp mảng tăng dần là: | 3 | |  |
| Cho mảng a gồm các phẩn tử có giá trị như sau: 1356  Số lần hoán vị 2 phần tử khác nhau khi áp dụng thuật toán nổi bọt để sắp xếp mảng giảm dần là: |  | | 6 |
| Cho mảng a gồm các phẩn tử có giá trị như sau: 3126  Số lần hoán vị 2 phần tử khác nhau khi áp dụng thuật toán nổi bọt để sắp xếp mảng giảm dần là: |  | | 4 |
| Cho mảng a gồm các phẩn tử có giá trị như sau: 3126  Số lần hoán vị 2 phần tử khác nhau khi áp dụng thuật toán nổi bọt để sắp xếp mảng giảm dần là: | 4 | |  |
| Cho mảng a gồm các phẩn tử có giá trị như sau: 74326  Số lần hoán vị 2 phần tử khác nhau khi áp dụng thuật toán chọn trực tiếp để sắp xếp mảng tăng dần là: | 3 | |  |
| Cho mảng a gồm các phẩn tử có giá trị như sau: 1356  Số lần hoán vị 2 phần tử khác nhau khi áp dụng thuật toán nổi bọt để sắp xếp mảng giảm dần là: |  | 6 |  |
|  |
| Cho mảng a gồm các phẩn tử có giá trị như sau: 3126  Số lần hoán vị 2 phần tử khác nhau khi áp dụng thuật toán đổi chỗ trực tiếp (Interchange Sort) để sắp xếp mảng tăng dần là: |  | 2 |  |
|  |
| Cho mảng a gồm các phẩn tử có giá trị như sau: 3126  Số lần hoán vị 2 phần tử khác nhau khi áp dụng thuật toán đổi chỗ trực tiếp (Bubble Sort) để sắp xếp mảng giảm dần là: |  | 4 |  |
|  |
| Cho mảng a gồm các phần tử: 8.3.2.7.6.4.2  Cho biết kết quả ở bước thứ 3 khi áp dụng thật toán sắp xếp Selection tăng dần trên mảng các phần tử trên | B: sai | | ??? |
| Cho mảng a gồm các phẩn tử có giá trị như sau: 3126  Số lần hoán vị 2 phần tử khác nhau khi áp dụng thuật toán đổi chỗ trực tiếp (Bubble Sort) để sắp xếp mảng giảm dần là: |  | | 4 |
|  |  | |  |
|  |  | |  |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Cho mảng a gồm các phẩn tử có giá trị như sau: 3126  Số lần hoán vị 2 phần tử khác nhau khi áp dụng thuật toán đổi chỗ trực tiếp (Interchange Sort) để sắp xếp mảng tăng dần là: | 2 | | | | | | | |
| Cho thuật toán sắp xếp Bubble Sort như sau:  void BubbleSort( int M[], int N)  {  for( int i = 0; i< N-1; i++) for( int j = N-1; j>I; j--)  if( M[j] <M[j-1]) Swap( M[j], M[j-1]); return ;  }  Chọn câu đúng nhất cho hàm Swap: | {  } | void Swap( int &X, int &Y) | | | | | |  |
|  | | | | | |
| int Temp = X; | | | | | |
| X=Y; |  | | | | |
| Y = Temp; | | | | | |
| return ; | | | | | |
|  | | | | | |
| Cho thuật toán sắp xếp Bubble Sort như sau:  void BubbleSort( int M[], int N)  {  for( int i = 0; i< N-1; i++) for( int j = N-1; j>I; j--)  if( M[j] <M[j-1]) Swap( M[j], M[j-1]); return ;  }  Chọn câu đúng nhất cho hàm Swap: |  | | | | | | | |
| {  } | void **Swap( int &X, int &Y)** | | | | |  | |
|  | | | | |
| int Temp = X; | | | |  |
| X=Y; |  | | |
| Y = Temp; | | |  |
| return ; | |  |
|  | |
| Cho thuật toán sau:  int LinenearSearch( int M[], int N, int X)  { int k = 0;  while (M[k] !=X && k<N) k++;  if (k<N) return k; return -1;  }  Chọn câu đúng nhất trong trường hợp xấu nhất khi không tìm thấy phần tử nào có giá trị bằng X: |  | Số phép gán: Gmax = 1 Số | | | | | |  |
| phép so sánh: Smax = 2N + | | | | | |
| 1 | | | | | |
| Cho thuật toán sau:  int LinearSearch( float M[], int N, float X)  { int k = 0; M[N] = X;  while (M[k] !=X)//n+1 k++;  if (k<N) return k; return -1;  }  Chọn câu đúng nhất trong trường hợp xấu nhất |  | Số phép gán: Gmax = 2 Số | | | | | |  |
| phép so sánh: Smax = N + 2 | | | | | |
|  | | | | | |

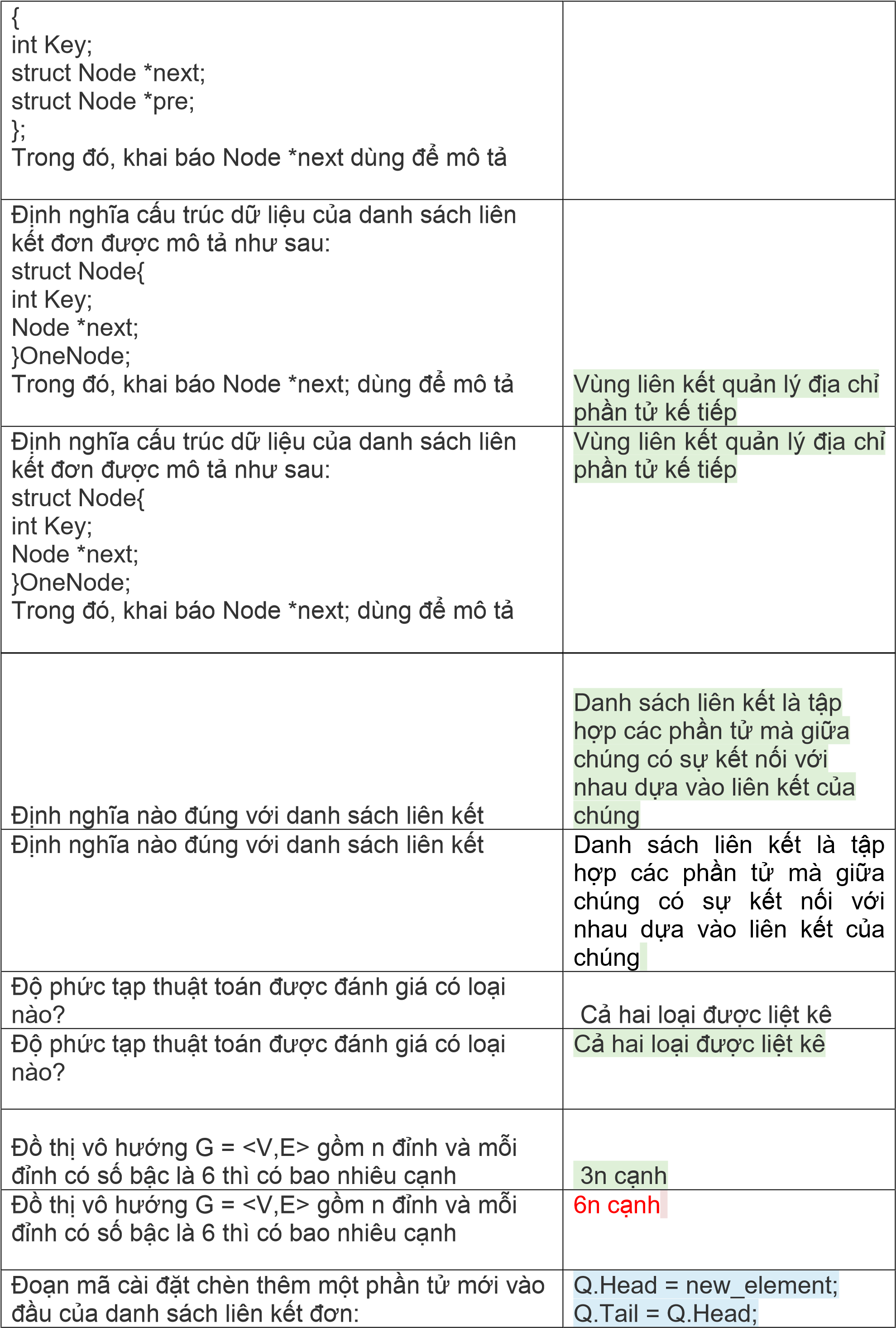
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| khi không tìm thấy phần tử nào có giá trị bằng X: |  | | | | |
| Cho thuật toán sau:  int LinearSearch( float M[], int N, float X)  { int k = 0; M[N] = X;  while (M[k] !=X)//n+1 k++;  if (k<N) return k; return -1;  }  Chọn câu đúng nhất trong trường hợp xấu nhất khi không tìm thấy phần tử nào có giá trị bằng X: |  | Số phép gán: Gmax = 2 Số | | |  |
| phép so sánh: Smax = N + 2 | | |
|  | | |
| Cho thuật toán sau:  int LinearSearch( float M[], int N, float X)  { int k = 0; M[N] = X;  while (M[k] !=X)//n+1 k++;  if (k<N) return k; return -1;  }  Chọn câu đúng nhất trong trường hợp xấu nhất khi không tìm thấy phần tử nào có giá trị bằng X: |  | | | | |
|  | Số phép gán: Gmax = 2 Số | |  | |
| phép so sánh: Smax = N + 2 | | |  |
| Cho thuật toán sau:  int LinenearSearch( int M[], int N, int X)  { int k = 0;  while (M[k] !=X && k<N) k++;  if (k<N) return k; return -1;  }  Chọn câu đúng nhất trong trường hợp xấu nhất khi không tìm thấy phần tử nào có giá trị bằng X: | Số phép gán: Gmax = 1 Số phép so sánh: Smax = 2N +  1 | | | | |
| Cho thuật toán tìm nhị phân không đệ quy sau:  int NrecBinarySearch( int M[], int N, int X)  {  int First = 0; int Last = N -1; while ( First <= Last)  {  int Mid = (First + Last)/2; |  | Số phép gán: Gmin = 3 Số | | |  |
| phép so sánh: Smin = 2 |  | |
|  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| if ( X == M[Mid]) return Mid; if ( X < M[Mid]) Last = Mid - 1; else First = Mid + 1;  }  return -1;  }  Chọn câu đúng nhất trong trường hợp tốt nhất khi phần tử ở giữa của mảng có giá trị bằng X: |  | | | | | |
| Cho thuật toán tìm nhị phân không đệ quy sau:  int NrecBinarySearch( int M[], int N, int X)  {  int First = 0; int Last = N -1; while ( First <= Last)  {  int Mid = (First + Last)/2; if ( X == M[Mid]) return Mid; if ( X < M[Mid]) Last = Mid - 1; else First = Mid + 1;  }  return -1;  }  Chọn câu đúng nhất trong trường hợp tốt nhất khi phần tử ở giữa của mảng có giá trị bằng X: |  | | | | | |
|  | Số phép gán: Gmin = 3 Số | | |  | |
| phép so sánh: Smin = 2 |  | |
| Danh sách được cài đặt bằng cách nào: |  | | | | | |
|  | Cả hai đáp án đều đúng |  | | | |
| Danh sách được cài đặt bằng cách nào: |  | Cả hai đáp án đều đúng |  | | | |
|  |
| Danh sách liên kết là gì? |  | là tập hợp các phần tử liên | | |  | |
| kết móc nối liên tiếp với |  | |
| nhau, có kiểu truy cập tuần | | |
| tự. Mỗi phần tử là một nút | |  |
| Danh sách liên kết là gì? |  | Cả hai phát biểu đều đúng | | |  | |
|  | | |
| Đâu là công thức tổng quát để tính giai thừa dựa vào giải thuật đệ quy | Giaithua = n\* Giaithua(n-1) | | | | | |
| Đâu là công thức tổng quát để tính giai thừa dựa vào giải thuật đệ quy | Giaithua = n\* Giaithua(n-  1) | | | | | |
| Đâu là định nghĩa của Hàng đợi |  | một kiểu danh sách trong | |  | | |
| đó được trang bị hai phép | |
| toán bổ sung một phần tử | |
| vào cuối danh sách và loại | | |  | |
| bỏ một phần tử ở đầu danh | | | |  |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | sách |  | | | | | |
|  |
| Đâu là định nghĩa của Hàng đợi |  | một kiểu danh sách trong đó | | | | | |  |
| được trang bị hai phép toán | | | | | |
| bổ sung một phần tử vào cuối danh sách và loại bỏ | | | | | |
| một phần tử ở đầu danh | | | | | |
| sách |  | | | | |
| Đâu là định nghĩa của Ngăn xếp | Dạng danh sách đặc biệt trong đó các phép toán thêm vào một phần tử mới hoặc loại bỏ một phần tử trong danh sách chỉ được phép thực hiện ở một đầu của danh sách | | | | | | | |
| Đâu là định nghĩa của Ngăn xếp |  | Dạng danh sách đặc biệt | | | | | |  |
| trong đó các phép toán thêm vào một phần tử mới | | | | | |
| hoặc loại bỏ một phần tử trong danh sách chỉ được | | | | | |
| phép thực hiện ở một đầu | | | | | |
| của danh sách | | | | |  |
| Đây là định nghĩa của độ phức nào? “Được tính là tổng số chi phí về mặt không gian (bộ nhớ) cần thiết sử dụng cho thuật toán” | Không gian | | | | | | | |
| Đây là định nghĩa của độ phức nào? “được tính là tổng số chi phí về mặt tổng thời gian cần thiết để hoàn thành thuật toán, được đánh giá dựa vào số lượng các thao tác được sử dụng trong thuật toán dựa trên bộ dữ liệu đầu vào | Thời gian | | | | | | | |
| Đây là định nghĩa của độ phức nào? “được tính là tổng số chi phí về mặt tổng thời gian cần thiết để hoàn thành thuật toán, được đánh giá dựa vào số lượng các thao tác được sử dụng trong thuật toán dựa trên bộ dữ liệu đầu vào  ” |  | Thời gian | |  | | | | |
|  | |
| Đây là định nghĩa của độ phức nào? “Được tính là tổng số chi phí về mặt không gian (bộ nhớ) cần thiết sử dụng cho thuật toán” |  | Không gian | | |  | | | |
|  | | |
| Để sắp xếp các phần tử của danh sách liên kết có mấy phương án sử dụng: |  | | | | | | | |
|  | 2 phương án | | | |  | | |
| Để sắp xếp các phần tử của danh sách liên kết có mấy phương án sử dụng: |  | 2 phương án | | | |  | | |
|  | | | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Để sắp xếp các phần tử của danh sách liên kết đôi sử dụng phương án nào? |  | | | | | | | | | | | | | |
|  | Cả hai phương án trên đều | | | | | | | | | | |  | |
| sai | |  | | | | | | | | |
| Để sắp xếp các phần tử của danh sách liên kết đôi sử dụng phương án nào? |  | Hoán vị nội dung của phần | | | | | | | | | | | |  |
| tử |  | | | | | | | | | | |
|  |
| Để sắp xếp các phần tử của danh sách liên kết đơn sử dụng phương án nào? |  | | | | | | | | | | | | | |
|  | Cả hai phương án trên đều | | | | | | | | | | |  | |
| đúng | | | |  | | | | | | |
| Để sắp xếp các phần tử của danh sách liên kết đơn sử dụng phương án nào? |  | Cả hai phương án trên đều | | | | | | | | | | | |  |
| đúng | | | |  | | | | | | | |
|  | | | |
| Để sử dụng hàm cấp phát bộ nhớ malloc(), calloc(), new(). Ta phải sử dụng thư viện nào? | stdlib.h | | | | | | | | | | | | | |
| Để sử dụng hàm cấp phát bộ nhớ malloc(), calloc(), new(). Ta phải sử dụng thư viện nào? |  | stdlib.h | | | | |  | | | | | | | |
|  | | | | |
| Để tiến hành tìm kiếm một phần tử trong danh sách liên kết đơn sử dụng phương pháp tìm kiếm gì? |  | | | | | | | | | | | | | |
|  | Tìm kiếm tuyến tính | | | | | | | | |  | | | |
| Để tiến hành tìm kiếm một phần tử trong danh sách liên kết đơn sử dụng phương pháp tìm kiếm gì? |  | Tìm kiếm tuyến tính | | | | | | | | |  | | | |
|  | | | | | | | | |
| Để tính biểu thức s = ½ + ¼ + … + 1/(2n) với n>=1 ta chọn hàm | float F( int n )  { if (n ==1 ) return 1.0/2; else  **return 1.0/2\*n + F(n-1)**;  } | | | | | | | | | | | | | |
|  | float F( int n ) | | | | | | | |  | | | | |
| { | | | | | | | |
| if (n ==1 ) | | | | | |  | |
| return 1.0/2; | | | | | | |  |
| else | | |  | | | |
| return 1.0/(2\*n) + F1(n-1); | | | | | | | | | |  | | |
| } | | | | | | | | | |
| Để tính biểu thức s = ½ + ¼ + … + 1/(2n) với n>=1 ta chọn hàm |  | float F( int n ) | | | | | | | | | | | |  |
| { | | | | | | | | | | | |
| if (n ==1 ) | | | | | | | | | | | |
| return 1.0/2; | | | | | | | | | | | |
| else | | |  | | | | | | | | |
| return 1.0/(2\*n) + F(n-1); | | | | | | | | | | | |
| } | | | | | | | | | | | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Để tính biểu thức s = ½ + 2/3 + ¾ + … + n/(n+1) ta chọn hàm | float F(int n)  {  if (n==1) return 1.0/2; else  **return (float)n/(n+1) + F(n1);**  } | | | | | | |
| Đoạn mã sau đây sử dụng thuật toán Sắp xếp gì?  void SXDSSV( int n, SV ds[]){ int min, i, j; SV tg; for( i=0 ; i<n-1 ; i++ ) { min = i; for( j=i+1 ; j<n ; j++ ) if ( ds[j].DTB < ds[min].DTB ) min = j; if( min != i )  { tg = ds[min];  ds[min] = ds[i];  ds[i] = tg; } }} | Selection sort | | | | | | |
| Để tính biểu thức s = ½ + 2/3 + ¾ + … + n/(n+1) ta chọn hàm | float F(int n)  {  if (n==1) return 1.0/2;  else  return (float)n/(n+1) + F(n-  1);  } | | | | | | |
| Để tính biểu thức s = xn với n>=0 ta chọn hàm | long F(int x, int n)  {  **if (n==0)** return 1; else  **return x\*F(x,n-1)**;  } | | | | | | |
| Để tính biểu thức s = xn với n>=0 ta chọn hàm | long F(int x, int n)  {  if (n==0) return 1;  else  return x\*F(x,n-1);  } | | | | | | |
| Để xác định giải thuật đệ quy cần xác định gì? | Cả hai lựa chọn đều đúng | | | | | | |
| Để xác định giải thuật đệ quy cần xác định gì? |  | Cả hai lựa chọn đều đúng | | |  | | |
|  | | |
| Định nghĩa cấu trúc dữ liệu của danh sách liên kết đôi được mô tả như sau:  struct Node  {  int Key; struct Node \*next; struct Node \*pre;  };  Trong đó, khai báo Node \*next dùng để mô tả |  | | | | | | |
|  | Vùng liên kết quản lý địa chỉ | | | |  | |
| phần tử kế tiếp của phần tử | | | |
| cuối |  | | |
| Định nghĩa cấu trúc dữ liệu của danh sách liên kết đôi được mô tả như sau:  struct Node |  | Vùng liên kết quản lý địa | | | | |  |
| chỉ phần tử kế tiếp | |  | | |
|  | |



|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| void insertFirst ( LIST &Q, Node \*new\_element ){ if ( Q.Head == NULL ) //nếu danh sách rỗng  {  [1] …….. [2] ……..  }  else//danh sách không rỗng  {  new\_element -> next = Q.Head;  Q.Head = new\_element;  }  }  Đoạn mã còn thiếu để đặt vào dòn số [1] và [2]. |  | | | | |  | |
| Đoạn mã sau đây thực hiện nhiệm vụ gì  void SXDSV\_InsertionSort( int n, SV ds[]){ int pos,i; SV x; for(i=1;i<n;i++)  { x = ds[i]; pos = i-1; while((pos>=0)&&(ds[pos].Tuoi>x.Tuoi)) { ds[pos+1] = ds[pos]; pos--; } ds[pos+1] = x; //chèn x vào dãy }} | Thực hiện sắp xếp danh sách SV theo Tuoi tăng dần bằng thuật toán Insertion | | | | |  | |
| Đoạn mã cài đặt chèn thêm một phần tử mới vào đầu của danh sách liên kết đơn: void insertFirst ( LIST &Q, Node \*new\_element ){ if ( Q.Head == NULL ) //nếu danh sách rỗng  {  Q.Head = new\_element;  Q.Tail = Q.Head;  }  else//danh sách không rỗng  {   1. …………… 2. ……………   }  }  Đoạn mã còn thiếu để đặt vào dòn số [1] và [2]. |  | | | | |  | |
|  | new\_element -> next = | |  | |
| Q.Head; |  |
| Q.Head = new\_element; | | |  |
| Đoạn mã cài đặt chèn thêm một phần tử mới vào đầu của danh sách liên kết đơn: void insertFirst ( LIST &Q, Node \*new\_element ){ if ( Q.Head == NULL ) //nếu danh sách rỗng  {  Q.Head = new\_element;  Q.Tail = Q.Head;  }  else//danh sách không rỗng  {   1. …………… 2. ……………   }  }  Đoạn mã còn thiếu để đặt vào dòn số [1] và [2]. |  | new\_element -> next | | | | = |  |
| Q.Head; |  | | |  |
| Q.Head = new\_element; | | |  |
|  | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Đoạn mã cài đặt chèn thêm một phần tử mới vào đầu của danh sách liên kết đơn:  void insertFirst ( LIST &Q, Node \*new\_element ){ if ( Q.Head == NULL ) //nếu danh sách rỗng  {  [1] …….. [2] ……..  }  else//danh sách không rỗng  {  new\_element -> next = Q.Head;  Q.Head = new\_element;  }  }  Đoạn mã còn thiếu để đặt vào dòn số [1] và [2]. |  | Q.Head = new\_element; | | |  |
| Q.Tail = Q.Head; | |  |
|  | |
| Đoạn mã cài đặt hàm tìm kiếm nhị phân phần tử x trên dãy sắp xếp tăng dần: int BinarySearch( int a[ ], int n, int x )  {  int left = ……….., right = ……………; int middle; do  {  middle = (left+right)/2; if (x == a[middle]) break;  else if (x<a[middle]) right = middle - 1; else left = middle + 1; } while ( left <= right ); if ( left <= right ) return middle;  else return -1;//ko tìm thấy phần tử x  }  Giá trị được điền vào dấu ………... để đoạn mã cài đặt thực hiện đúng: | 0 và n-1 | | | | |
| Đoạn mã cài đặt hàm tìm kiếm nhị phân phần tử x trên dãy sắp xếp tăng dần:  int BinarySearch( int a[ ], int n, int x )  {  int left = ……….., right = ……………; int middle; do  {  middle = (left+right)/2; if (x == a[middle]) break;  else if (x<a[middle]) right = middle - 1; else left = middle + 1; } while ( left <= right ); |  | 0 và n-1 |  | | |
|  |

|  |  |  |  |
| --- | --- | --- | --- |
| if ( left <= right ) return middle;  else return -1;//ko tìm thấy phần tử x  }  Giá trị được điền vào dấu ………... để đoạn mã cài đặt thực hiện đúng: |  | | |
| Đoạn mã cài đặt hủy bỏ một phần tử đứng sau một phần tử q trong danh sách liên kết đơn: void RemoveAfter ( LIST &Q , Node \*q ){ Node \*p;  if (q != NULL)  {  p = q -> next;  if (p != NULL)  {  if (p == Q.Tail)  {  q->next = NULL;  Q.Tail = q;  }  [1] ………………….  free(p);  }  }  else RemoveHead(Q);  }  Dòng lệnh cần thiết được đặt vào chỗ trống tại dòng số [1]: |  | | |
|  | q -> next = p -> next; |  |
| Đoạn mã cài đặt hủy bỏ một phần tử đứng sau một phần tử q trong danh sách liên kết đơn: void RemoveAfter ( LIST &Q , Node \*q ){ Node \*p; if (q != NULL)  {  p = q -> next;  if (p != NULL)  {  if (p == Q.Tail)  {  q->next = NULL;  Q.Tail = q;  }  [1] ………………….  free(p);  }  } |  | q -> next = p -> next; |  |
|  |

|  |  |  |  |
| --- | --- | --- | --- |
| else RemoveHead(Q);  }  Dòng lệnh cần thiết được đặt vào chỗ trống tại dòng số [1]: |  | | |
| Đoạn mã cài đặt hủy phần tử đầu của danh sách liên kết đơn:  void RemoveHead ( LIST &Q ){ Node \*p;  if (Q.Head != NULL)  {  p = Q.Head;  [1] …………………..  free(p);  if ( Q.Head == NULL )  Q.Tail = NULL;  }  }  Dòng lệnh cần thiết được đặt vào chỗ trống tại dòng số [1]: |  | | |
|  | Q.Head = Q.Head -> next; |  |
| Đoạn mã cài đặt hủy phần tử đầu của danh sách liên kết đơn:  void RemoveHead ( LIST &Q ){ Node \*p;  if (Q.Head != NULL)  {  p = Q.Head;  [1] …………………..  free(p);  if ( Q.Head == NULL )  Q.Tail = NULL;  }  }  Dòng lệnh cần thiết được đặt vào chỗ trống tại dòng số [1]: | T={(1, 2), (1, 4), (2, 4), (2, 6), (4, 5), (6, 7)} | | |
| Đoạn mã cài đặt hủy phần tử đầu của danh sách liên kết đơn: void RemoveHead ( LIST &Q ){ Node \*p;  if (Q.Head != NULL)  {  p = Q.Head;  [1] …………………..  free(p);  if ( Q.Head == NULL )  Q.Tail = NULL; |  | Q.Head = Q.Head -> next; |  |
|  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| }  }  Dòng lệnh cần thiết được đặt vào chỗ trống tại dòng số [1]: |  | | | |
| Đoạn mã để tạo ra nút mới có thành phần là x trong danh sách liên kết đôi với mỗi nút gồm các thành phần (infor, next, pre) sau:  Node\* get\_node( Data x ){  Node \*p;  p = (Node\*)malloc(sizeof(Node));  if ( p == NULL )  {  printf(“Ko du bo nho”);  exit(1);  }  p ->infor = x; p -> …. = NULL; p -> pre = NULL;  return p;  }  Điền phần còn thiếu vào chỗ …………. |  | | | |
|  | next | |  |
| Đoạn mã để tạo ra nút mới có thành phần là x trong danh sách liên kết đơn với mỗi nút gồm hai thành phần (infor, next) sau:  Node\* get\_node( Data x ){  Node \*p;  p = (Node\*)malloc(sizeof(Node));  if ( p == NULL )  {  printf(“Ko du bo nho”);  exit(1);  }  p -> infor = ……; p -> next = NULL;  return p;  }  Điền phần còn thiếu vào chỗ ………….. |  | | | |
|  | X |  | |
| Đoạn mã để tạo ra nút mới có thành phần là x trong danh sách liên kết đơn với mỗi nút gồm hai thành phần (infor, next) sau:  Node\* get\_node( Data x ){  Node \*p;  p = (Node\*)malloc(sizeof(Node));  if ( p == NULL )  { |  | | | |
|  | next | |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| printf(“Ko du bo nho”);  exit(1);  }  p -> infor = x; p -> ….. = NULL;  return p;  }  Điền phần còn thiếu vào chỗ ………….. Đoạn mã để tạo ra nút mới có thành phần là x trong danh sách liên kết đôi với mỗi nút gồm các thành phần (infor, next, pre) sau:  Node\* get\_node( Data x ){  Node \*p;  p = (Node\*)malloc(sizeof(Node));  if ( p == NULL )  {  printf(“Ko du bo nho”);  exit(1);  }  p -> …….. = x; p -> next = NULL; p -> pre = NULL;  return p;  }  Điền phần còn thiếu vào chỗ ………….. infor |  | | | | |
| Đoạn mã để tạo ra nút mới có thành phần là x trong danh sách liên kết đôi với mỗi nút gồm các thành phần (infor, next, pre) sau:  Node\* get\_node( Data x ){ Node \*p;  ……………………..  if ( p == NULL )  {  printf(“Ko du bo nho”);  exit(1);  }  p -> infor = x; p -> next = NULL; p -> pre = NULL;  return p;  }  Điền phần còn thiếu vào chỗ ………….. |  | | | | |
|  | p = | |  | |
| (Node\*)malloc(sizeof(Node)) | | |  |
| ; | | |
| Đoạn mã để tạo ra nút mới có thành phần là x trong danh sách liên kết đơn với mỗi nút gồm hai thành phần (infor, next) sau:  Node\* get\_node( Data x ){  Node \*p; |  | x |  | | |
|  |

|  |  |  |  |
| --- | --- | --- | --- |
| p = (Node\*)malloc(sizeof(Node));  if ( p == NULL )  {  printf(“Ko du bo nho”);  exit(1);  }  p -> infor = ……; p -> next = NULL;  return p;  }  Điền phần còn thiếu vào chỗ ………….. |  | | |
| Đoạn mã để tạo ra nút mới có thành phần là x trong danh sách liên kết đơn với mỗi nút gồm hai thành phần (infor, next) sau:  Node\* get\_node( Data x ){  Node \*p;  p = (Node\*)malloc(sizeof(Node));  if ( p == NULL )  {  printf(“Ko du bo nho”);  exit(1);  }  p -> infor = x; p -> ….. = NULL;  return p;  }  Điền phần còn thiếu vào chỗ ………….. |  | next |  |
|  |
| Đoạn mã để tạo ra nút mới có thành phần là x trong danh sách liên kết đôi với mỗi nút gồm các thành phần (infor, next, pre) sau:  Node\* get\_node( Data x ){  Node \*p;  p = (Node\*)malloc(sizeof(Node));  if ( p == NULL )  {  printf(“Ko du bo nho”);  exit(1);  }  p -> …….. = x; p -> next = NULL; p -> pre = NULL;  return p;  }  Điền phần còn thiếu vào chỗ ………….. |  | infor |  |
|  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Đoạn mã để tạo ra nút mới có thành phần là x trong danh sách liên kết đôi với mỗi nút gồm các thành phần (infor, next, pre) sau:  Node\* get\_node( Data x ){ Node \*p;  ……………………..  if ( p == NULL )  {  printf(“Ko du bo nho”);  exit(1);  }  p -> infor = x; p -> next = NULL; p -> pre = NULL;  return p;  }  Điền phần còn thiếu vào chỗ ………….. |  | p = | | |  |
| (Node\*)malloc(sizeof(Node)) | | |
| ; | | |
| Đoạn mã dưới đây mô tả thuật toán gì: B1: k = 1  B2: if M[k] == X and k !=n B2.1: k++  B2.2: Lặp lại bước 2  B3: if (k<N) thông báo tìm thấy tại vị trí thứ k  B4: else thông báo không tìm thấy  B5: Kết thúc |  | | | | |
|  | Tìm kiếm tuyến tính phần tử | | |  |
| X trong mảng | |  |
| Đoạn mã dưới đây mô tả thuật toán gì: B1: k = 1  B2: if M[k] == X and k !=n B2.1: k++  B2.2: Lặp lại bước 2  B3: if (k<N) thông báo tìm thấy tại vị trí thứ k  B4: else thông báo không tìm thấy  B5: Kết thúc |  | Tìm kiếm tuyến tính phần tử | | |  |
| X trong mảng | |  |
|  | |
| Đoạn mã khởi tạo danh sách rỗng sau: void init( List &Q ){  Q.Head = ......;  Q.Tail = NULL;  }  Phần còn thiếu điền vào dấu ……. là gì |  | | | | |
|  | NULL |  | | |
| Đoạn mã khởi tạo danh sách rỗng sau: void init( List &Q ){  Q.Head = ......;  Q.Tail = NULL;  }  Phần còn thiếu điền vào dấu ……. là gì |  | NULL |  | | |
|  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Đoạn mô tả này thuộc thuật toán nào:  Bước 1: i = 0  Bước 2: tính các giá trị j = i + 1  Bước 3: Trong khi j<n thực hiện   * nếu a[j] < a[i] thì hoán đổi a[i] với a[j] * j = j + 1;   Bước 4: i = i +1 nếu i<n-1 thì lặp lại bước 2, ngược lại thì dừng |  | Sắp xếp đổi chỗ trực tiếp | | |  | |
|  | | |
| Đoạn mô tả này thuộc thuật toán nào:  Bước 1: i = 0  Bước 2: tính các giá trị j = i + 1  Bước 3: Trong khi j<n thực hiện   * nếu a[j] < a[i] thì hoán đổi a[i] với a[j] * j = j + 1;   Bước 4: i = i +1  nếu i<n-1 thì lặp lại bước 2, ngược lại thì dừng |  | | | | | |
|  | Sắp xếp đổi chỗ trực tiếp | | |  | |
| Đối với thuật toán sắp xếp chọn trực tiếp cho dãy phần tử sau (10 phần tử): 16 60 2 25 15 45 5 30 33 20  Cần thực hiện bao nhiêu lựa chọn phần tử nhỏ nhất để sắp xếp mảng M có thứ tự tăng dần | 9 lần | | | | | |
| Đối với thuật toán sắp xếp chọn trực tiếp cho dãy phần tử sau (10 phần tử): 16 60 2 25 15 45 5 30 33 20  Cần thực hiện bao nhiêu lựa chọn phần tử nhỏ nhất để sắp xếp mảng M có thứ tự tăng dần |  | 9 lần |  | | | |
|  |
| Giả sử cần sắp xếp mảng M có N phần tử sau theo phưuơng pháp sắp xếp chèn trực tiếp: 11 16 12 75 51 54 5 73 36 52 98  Cần thực hiện bao nhiêu lần chèn các phần tử vào dãy con đã có thứ tự tăng dần đứng đầu dãy M để sắp xếp mảng tăng dần: | 10 lần | | | | | |
| Giả sử cần sắp xếp mảng M có N phần tử sau theo phưuơng pháp sắp xếp chèn trực tiếp: 11 16 12 75 51 54 5 73 36 52 98  Cần thực hiện bao nhiêu lần chèn các phần tử vào dãy con đã có thứ tự tăng dần đứng đầu dãy M để sắp xếp mảng tăng dần: |  | 10 lần | |  | | |
|  | |
| Giả sử T = <V, > là đồ thị n đỉnh. Khẳng định nào không tương đương với các khẳng định còn lại |  | | | | | |
|  | T có đúng một chu trình n-1 | | | |  |
| cạnh |  | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Hàm mô tả sắp xếp nổi bọt (Bubble Sort) trên mảng M có N phần tử:   1. void BubbleSort(int M[ ], int N) 2. {   3.int i,j,tg;  4.for( i = 0 ; i < N-1 ; i++ )  5.........................................  6.if ( M[j] < M[j-1] )  7.{  8.tg = M[j];   1. M[ j] = M[j-1]; 2. M[ j-1] = tg;   11.}  12.}  Lệnh nào sau đây s được đưa vào dòng số [5] của đoạn mã trên |  | | | | |
|  | for( j = N-1; j>i; j--) | | |  |
| Hàm mô tả sắp xếp nổi bọt (Bubble Sort) trên mảng M có N phần tử:   1. void BubbleSort(int M[ ], int N) 2. {   3.int i,j,tg;  4.for( i = 0 ; i < N-1 ; i++ )  5.........................................  6.if ( M[j] < M[j-1] )  7.{  8.tg = M[j];   1. M[ j] = M[j-1]; 2. M[ j-1] = tg;   11.}  12.}  Lệnh nào sau đây s được đưa vào dòng số [5] của đoạn mã trên |  | for( j = N-1; j>i; j--) | |  | |
|  | |
| Hàng đợi còn được gọi là danh sách | FIFO | | | | |
| Hàng đợi còn được gọi là danh sách |  | FIFO |  | | |
|  |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Hãy cho biết đồ thị nào đưới đây là một cây |  | | | | | | | | | |
|  | Phương án C |  | | | | | | | |
| Hãy cho biết đồ thị nào đưới đây là một cây | Phương án c | | | | | | | | | |
| Lựa chọn câu đúng nhất về danh sách liên kết đôi. |  | Vùng liên kết của một phần | | | | | | |  | |
| tử trong danh sách liên kết | | | | | | |
| đôi có 02 mối liên kết vớ | | | | | | i |
| phần tử trước và sau nó | | | | |  |
| trong danh sách | |  | | |
| Lựa chọn phương án trả lời đúng nhất cho biết hình ảnh sau là gì? |  | Cây nhị phân tìm kiếm | | | |  | | | | |
|  | | | |
| Ma trận kề của đồ thị có hướng G = <V, > |  | | | | | | | | | |
|  | Là ma trận không đối xứng | | | | | | |  | |
| Ma trận kề của đồ thị có hướng G = <V, > |  | Là ma trận không đối xứng | | | | | | |  | |
|  | | | | | | |
| Ma trận kề của đồ thị vô hướng G =<V, > có tính chất |  | | | | | | | | | |
|  | Là ma trận đường chéo trên | | | | | | | |  |
| Ma trận kề của đồ thị vô hướng G =<V, > có tính chất |  | Là ma trận đối xứng | | |  | | | | | |
|  | | |
| Ma trận kề nào dưới đây biểu diễn đúng của đồ thị trong số đã cho trong hình v | sai | | | | | | | | | |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Ma trận kề nào dưới đây biểu diễn đúng của đồ thị trong số đã cho trong hình v |  | | | | | | | | | |
| Mỗi nút trong danh sách đơn gồm có mấy phần: |  | | | | | | | | | |
|  | 2 phần | | | |  | | | | |
| Mỗi nút trong danh sách đơn gồm có mấy phần: |  | 2 phần | | |  | | | | | |
|  | | |
| Một chương trình cài đặt trên máy tính được xác định bởi thành phần nào | Cả hai thành phần | | | | | | | | | |
| Một chương trình cài đặt trên máy tính được xác định bởi thành phần nào |  | Cả hai thành phần | | | | |  | | | |
|  | | | | |
| Ngăn xếp còn được gọi là danh sách |  | | | | | | | | | |
|  | LIFO | |  | | | | | | |
| Ngăn xếp còn được gọi là danh sách |  | LIFO |  | | | | | | | |
|  |
| Phần tử thế mạng có thể được dùng khi xóa nút trong trường hợp nút có hai nhánh con là gì? | Cả hai phát biểu đều đúng | | | | | | | | | |
| Phần tử thế mạng có thể được dùng khi xóa nút trong trường hợp nút có hai nhánh con là gì? |  | Cả hai phát biểu đều đúng | | | | | | |  | |
|  | | | | | | |
| Phương pháp duyệt NLR là phương pháp duyệt gì? |  | | | | | | | | | |
|  | Node - Left - Right | | | | |  | | | |
| Phương pháp duyệt NLR là phương pháp duyệt gì? |  | Node - Left - Right | | | | |  | | | |
|  | | | | |
| Ta gọi đỉnh v là đỉnh cô lập trong đồ thị vô hướng G = <V, E> | Nếu bậc của đỉnh v là một số lẻ | | | | | | | | | |
| Ta gọi đỉnh v là đỉnh cô lập trong đồ thị vô hướng G = <V, E> |  | Nếu bậc của đỉnh v là một | | | | | | | |  |
| số chẵn | | | |  | | | |
|  | | | |
| Ta gọi đỉnh v là đỉnh treo trong đồ thị vô hướng G = <V, E> |  | | | | | | | | | |
|  | Nếu bậc của đỉnh v là 1 | | | | | |  | | |
| Ta gọi đỉnh v là đỉnh treo trong đồ thị vô hướng G = <V, E> |  | Nếu bậc của đỉnh v là 1 | | | | | |  | | |
|  | | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Thao tác thêm một phần tử vào cây khi so sánh giá trị của phần tử cần thêm vào so với nút đang xét nếu phần tử cần thêm vào lớn hơn thì được thêm vào vị trí nào? | Phần tử mới được bổ sung vào nhánh trái của nút đang xét | | | | |
| Thao tác thêm một phần tử vào cây khi so sánh giá trị của phần tử cần thêm vào so với nút đang xét nếu phần tử cần thêm vào lớn hơn thì được thêm vào vị trí nào? |  | Phần tử mới được bổ sung | | |  |
| vào nhánh trái của nút đang | | |
| xét |  | |
|  |
| Thủ tục mô tả thuật toán sắp xếp chọn trực tiếp: void SapXepChonTrucTiep( T M[], int N)  {  int K = 0, posmin; int Temp;  ................................................  {  T Min = M[K]; Posmin = K;  for( int pos = K+1; pos<N; pos++)  if( Min > M[pos])  {  Min = M[pos];  Posmin = pos;  }  Temp = M[k];  M[k] = m[posmin];  M[posmin] = Temp;  }  return;  }  Đoạn mã cần thiết để đặt vào dòng  .....................để chương trình sắp xếp đúng |  | for ( k =0; k<n-1; k++) | |  | |
|  | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Thủ tục mô tả thuật toán sắp xếp chọn trực tiếp: void SapXepChonTrucTiep( T M[], int N)  {  int K = 0, posmin; int Temp;  ................................................  {  T Min = M[K]; Posmin = K;  for( int pos = K+1; pos<N; pos++)  if( Min > M[pos])  {  Min = M[pos];  Posmin = pos;  }  Temp = M[k];  M[k] = m[posmin];  M[posmin] = Temp;  }  return;  }  Đoạn mã cần thiết để đặt vào dòng  .....................để chương trình sắp xếp đúng |  | | | | | | | |
|  | for ( k =0; k<n-1; k++) | |  | | | | |
| Thuật toán được biểu diễn bằng cách nào | Tất cả các cách được liệt kê | | | | | | | |
| Thuật toán được biểu diễn bằng cách nào |  | Tất cả các cách được liệt kê | | | | | |  |
|  | | | | | |
| Thuật toán được biểu diễn bằng cách nào |  | Tất cả các cách được liệt kê | | | | | |  |
|  | | | | | |
| Tổ chức của danh sách liên kết kép gồm có mấy thành phần: |  | 3 thành phần |  | | | | | |
|  |
| Tổng các phần tử hàng i, cột j của ma trận kề đồ thị có hướng G =<V, > đúng bằng: |  | | | | | | | |
|  | Bán đỉnh bậc ra của đỉnh i, | | | | |  | |
| bán đỉnh bậc vào đỉnh j. | | |  | |
| Tổng các phần tử hàng i, cột j của ma trận kề đồ thị có hướng G =<V, > đúng bằng: |  | Bán đỉnh bậc ra của đỉnh i | | | | | | , |
| bán đỉnh bậc vào đỉnh j. | | |  | | |
|  | | |
| Tổng các phần tử hàng i, cột j của ma trận kề đồ thị vô hướng G = <V, > đúng bằng |  | | | | | | | |
|  | Bậc của đỉnh i, đỉnh j | |  | | | | |
| Tổng các phần tử hàng i, cột j của ma trận kề đồ thị vô hướng G = <V, > đúng bằng | Bậc của đỉnh i, đỉnh j | | | | | | | |
| Tổng các phần tử trên hàng trong ma trận kề của đồ thị có hướng G = <V, > đúng bằng |  | | | | | | | |
|  | Cả ba phương án đều sai | | | |  | | |

Tổng các phần tử trên hàng trong ma trận kề của đồ thị có hướng G = <V, > đúng bằng

Cả ba phương án đều sai

Tổng các phần tử trên một hàng hoặc của một cột trong ma trận kề của đồ thị vô hướng G = Số cạnh liên thuộc với đỉnh

<V, > đúng bằng của cột hoặc hàng đó

Tổng các phần tử trên một hàng hoặc của một cột trong ma trận kề của đồ thị vô hướng G = Số cạnh liên thuộc với đỉnh

<V, > đúng bằng của cột hoặc hàng đó

Tổng các phần tử trên một hàng hoặc của một Số cạnh liên thuộc với đỉnh cột trong ma trận kề của đồ thị vô hướng G = của cột hoặc hàng đó <V, > đúng bằ

Trong đồ thị vô hướng, số đỉnh bậc lẻ là một số Chia hết cho 2

Trong đồ thị vô hướng, số đỉnh bậc lẻ là một số Chia hết cho 2

Trong giải thuật đệ quy thì lời giải trực tiếp mà

không phải nhờ đến một bài toán con nào đó là

thành phần nào? Phần tử neo

Trong giải thuật đệ quy thì lời giải trực tiếp mà Phần tử neo không phải nhờ đến một bài toán con nào đó là thành phần nào?

Để lưu trữ địa chỉ của nút kế tiếp hoặc giá trị NULL

Trong một nút của danh sách liên kết đơn, thành nếu không liên kết đến phần phần infor là thành phần gì? tử nào

Trong một nút của danh sách liên kết đơn, thành Để lưu trữ địa chỉ của nút kế phần infor là thành phần gì? tiếp hoặc giá trị NULL nếu

không liên kết đến phần tử nào

Tất cả các phương án đều

Ứng dụng cơ bản của ngăn xếp gồm đúng

Ứng dụng cơ bản của ngăn xếp gồm Tất cả các phương án đều

void RemoveHead ( LIST &Q ){ Node \*p; if (Q.Head != NULL)

{

p = Q.Head;

…[1] …

free(p); if ( Q.Head == NULL )

Q.Tail = NULL;

}

đúng

Q.Head = Q.Head -> next;

|  |  |  |  |
| --- | --- | --- | --- |
| }  Dòng lệnh cần thiết được đặt vào chỗ trống tại dòng số [1]: | | |  |
|  |  |  |  |
|  |

|  |  |
| --- | --- |
|  | In đầy đủ thông tin tất cả các cán bộ đang chứa trong danh sách Q lần lượt từ chối danh sách về đầu danh sách |
| Cho Danh sách liên kết đơn chứa danh sách Cán Bộ (CB), Đoạn mã sau đây thực hiện gì?  void InDSCanBo (List Q)  {  Node \*p;  for(p=Q.Head; p!=NULL; p=p->next)  {  System.out.print(“%5d”, p->info.mcb);  System.out.print(“%15s”, p->info.hoten);  System.out.print(“%10s”, p->info.ns);  System.out.print(“%7.1f”, p->info.hsl);  System.out.print(“%7.0f”, p->info.pc);  }  } | I n đầy đủ thông tin tất cả các cán bộ đang chứa trong danh sách Q |
| Đoạn mã sau đây thực hiện yêu cầu xử lý gì?  int TK\_SV\_Ten\_Tuyentinh(int n, SV ds[], char ht[]){ for(int i = 0; i<n; i++) if ( strcmp(ds[i].HoTen, ht)==0) break; if (i<n) return 1; else return -1;} | Thực hiện tìm kiếm trong danh sách (ds) có SV có tên là ht hay không? Nếu có thì trả lại giá trị là 1, ngược lại không có trả lại giá trị là -1, thuật toán sử dụng là thuật toán tìm kiếm tuyến tính |