I. Trắc nghiệm

Câu 1:

Kết quả in ra màn hình là D. 6

Giải thích : toán tử điều kiện ?:

Điều kiện x > 0 đúng ( 2>0)

-> trả về 1 ; a += 1 -> a = 6

Câu 2 :

Số lượng bước cần thiết để chuyển 3 đĩa từ A sang C là C . 7

Giải thích : tính theo cthuc : T(n)= 2T(n-1) + 1

VD : T(1) = 1 ( 1 bước để di chuyển 1 đĩa)

T(2) = 2(1) + 1 = 3

T(3) = 2(3) + 1 = 7

Chi tiết : n là số lượng đĩa cần di chuyển , T(n) là số bước tối thiểu cần để di chuyển toàn bộ n đĩa từ cọc nguồn sang cọc đích theo quy tắc của bài Toán tháp HN : -chỉ có thể di chuyển từng đĩa 1, ko đc đặt đĩa lớn lên đĩa nhỏ

Tương đương cthuc : 2^n – 1 cũng ra kqua đúng

Câu 3 : Khi gọi printArray(arr, 5) với arr là {1, 2, 3, 4, 5} , kết quả sẽ là A) 1 2 3 4 5

Giari thích : in ra dãy theo vòng lặp for từ 0 -> size

Câu 4 : Giả sử có một mảng arr = {5, 1, 4, 2, 8} . Bạn được yêu cầu sắp xếp mảng tăng dần bằng phương pháp nổi bọt, sau lần sắp xếp đầu tiên, mảng sẽ trông C) {1, 4, 2, 5, 8}

Explain : sắp xếp nổi bọt là so sánh coi cái nào lớn hơn thì để ra sau, so sánh lần lượt trong lần 1 theo thứ tự ( 5 vs 1, 4, 2, 8)

Câu 5 :

Điều kiện của vòng lặp nào sau đây sẽ làm vòng lặp chạy vô hạn?

x = -1;

while (condition) {

// code A) condition = true

}

Câu 6 : Giá trị của factorial(4) là C) 24   
Explain : hàm đệ quy tính giai thừa 4! = 1.2.3.4= 24

Câu 7 : Hàm tìm kiếm nhị phân trả lại vị trí của key trong mảng hoặc trả về -1. Giả sử có một mảng đã sắp xếp arr = {1, 3, 5, 7, 9} . Chúng ta cần tìm kiếm giá trị 2 . Giá trị trả về của hàm tìm kiếm nhị phân là C) -1

Câu 8 : void changeValue(int &x)

{ x = 10; }

Nếu gọi hàm changeValue(a) với a là 5 , giá trị của a sau khi gọi hàm sẽ là A) 5

Explain( chống đào lửa ) : chỗ này là hàm kia với a = 5 ở trong, và hàm truyền tham trị (có thể thay đổi giá trị sau khi gọi, khác với truyền tham số chỉ là copy) -> a = 10

Câu 9:

void incrementByValue(int x) { x++; }

void incrementByReference(int &x) { x++; }

Nếu gọi hàm incrementByValue(a) và incrementByReference(b) với a và b đều là 5 , giá trị của a và b sau khi gọi hàm sẽ lần lượt là C) 5, 6  
Explain : đơn giản là truyền tham số ( đầu tiên ) thì chỉ copy thôi chứ ko đổi được giá trị khi gọi hàm còn truyền tham trị thì có khả năng đó

Câu 10 :

int fibonacci(int n) {

if (n <= 1)

return n;

else

return fibonacci(n - 1) + fibonacci(n - 2);

} Giá trị của fibonacci(6) là B) 8

Explain : dãy fibonacci – special đấy ( 0 1 1 2 3 5 8 13 21 34…..)

-> số thứ 6 là 8 (mà cũng khoai)

II. Tự luận

Câu 3 : Bubble sort hay còn gọi là sắp xếp nổi bọt là thuật toán sắp xếp đơn giản, hoạt động bằng cách lặp đi lặp lại qua danh sách cần sắp xếp, so sánh từng cặp phần tử liền kề và hoán đổi chúng nếu chúng nằm sai thứ tự.

\*Cách hoạt động : - Duyệt qua mảng nhiều lần

- Trong mỗi lần duyệt, so sánh từng cặp phần tử liên tiếp:

+) nếu phần tử trước lớn hơn phần tử sau -> hoán đổi chúng

- Sau mỗi lần duyệt, phần tử lớn nhất ( or nhỏ nhất) sẽ nổi lên đúng chỗ

- Quá trình lặp lại cho đến khi mảng được sắp xếp hoàn toàn.

\*Về ưu- nhược điểm bản thân Sắp xếp nổi bọt :

Ưu : Dễ triển khai – Không yêu cầu bộ nhớ bổ sung – hoạt động tốt với mảng kích

thước nhỏ

Nhược : Chậm với dữ liệu lớn, kém so với các cách sắp xếp khác, tốn tài nguyên vì

biến đổi nhiều

Khi so sánh với insert và select ( sắp xếp) :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tên | Độ phức tạp | Tốt cho mảng nhỏ | ỔN ĐỊNH? | Đặc điểm |
| Nổi bọt | O(n2) | YES | YES | So sánh từng cặp và hoán đổi |
| Lựa chọn | O(n2) | YES | NOPE | Tính phần tử nhỏ nhất rồi đổi chỗ |
| Chèn | O(n2), tốt nhất O(n) | The Best | YES | Xây dựng mảng sắp xếp từng bước |