[부스트코스] 모두를 위한 컴퓨터 과학 첫 걸음 퀴즈 해설

4. 알고리즘

boostcourse

Quiz 1

알고리즘의 성능 및 시간 복잡도를 표현하는 표기법 중 하나로, 최악의 경우일때(상한)를 나타내는 것은 다음 중 무엇인가요?

O()

Ω()

θ()

φ()

-> Big-O: O()는 상한/최악의 경우, Big-Omega: Ω()는 하한/최선의 경우, Big-Theta: θ()는 상한과 하한의 평균적인 시간 복잡도가 같을때를 의미합니다. Phi: φ()는 시간 복잡도 표기법에서 미리 합의된 의미가 없습니다.

boostcourse

Quiz 2

name과 number 두개의 멤버를 갖는 person이라는 새로운 자료형을 구조체로 정의하고자 합니다. 아래 코드의 괄호 안에 들어갈 코드로 알맞은 것은 무엇인가요? (코드 생략)

typedef struct

function struct

construct

function

-> 사용자 정의 자료형를 정의는 방법은 typedef struct { ... } 입니다. typedef는 자료형을 재정의할 때 사용하는 키워드이며, struct는 구조체를 의미합니다.

boostcourse

Quiz 3

전화번호부 책에서 '이펭수'를 찾는 작업을 선형 검색으로 수행하게 될 경우 Big-O 는 어떻게 될까요?

O(1)

O(log n)

O(n)

O(n^2)

-> 전화번호부 책을 맨 앞장에서 부터 차근차근 검색하는 선형 검색의 경우 100페이지라면 100번, 500 페이지라면 500번 작업을 반복하게 되므로 Big-O는 O(n)이 됩니다.

boostcourse

Quiz 4

5 6 7 3 2 과 같은 숫자 리스트가 주어졌습니다. 오름차순 정렬을 위해 버블 정렬을 왼쪽 처음부터 오른쪽 끝까지 '한 번' 수행했을 때의 리스트는 어떻게 될까요?

5 6 3 2 7

2 3 5 6 7

5 6 7 2 3

5 6 2 3 7

-> 앞에서부터 각 숫자 쌍을 보면서 정렬이 필요할 경우 순서를 바꿉니다. 5와 6은 잘 정렬되어 있으므로 그대로 둡니다. 6과 7은 잘 정렬이 되어 있으므로 그대로 둡니다. 7과 3은 순서를 바꿉니다. (순서를 바꾼 후 다음 쌍인) 7과 2도 순서를 바꿉니다. 따라서 수정된 리스트는 5 6 3 2 7이 됩니다.

Quiz 5

5 6 7 3 2 와 같은 숫자 리스트가 주어졌습니다. 오름차순 정렬을 위해 선택 정렬을 통해 교환을 '한 번' 수행했을 때의 리스트는 어떻게 될까요?

for (int i = 1; i < 10; i++) {... scores[i] ...}

for (int i = 0; i < 10; i++) {... scores[i] ...}

for (int i = 1; i < 11; i++) {... scores[i] ...}

for (int i = 0; i < 11; i++) {... scores[i] ...}

-> 배열을 정의한 코드로부터 배열의 크기가 10이라는 것을 알 수 있습니다. for 루프를 사용해서 배열의 인덱스인 0부터 9까지 순환하는 코드를 작성해야 합니다. 따라서 i는 0부터 시작해서 1씩 증가시켜 가면서 10보다 작아야 합니다.

boostcourse

Quiz 6

선택 정렬, 버블 정렬, 선형 검색, 이진 검색 4가지 알고리즘이 최선인 경우일 때의 실행시간이(하한) 빠른 순서대로 나열한 것은 무엇인가요? (단, 하한이 같은 경우 상한이 빠른 순으로 나열합니다)

선택 정렬 - 버블 정렬 - 선형 검색 - 이진 검색

버블 정렬 - 선택 정렬 - 선형 검색 - 이진 검색

선형 검색 - 이진 검색 - 선택 정렬 - 버블 정렬

이진 검색 - 선형 검색 - 버블 정렬 - 선택 정렬

-> 4가지 알고리즘의 하한은 다음과 같습니다. Ω(1): 선형 검색, 이진 검색 < Ω(n): 버블 정렬 < Ω(n^2): 선택 정렬 여기서 하한이 같은 경우를 위해 상한을 비교하면 다음과 같습니다. O(log n): 이진 검색 < O(n): 선형 검색 < O(n^2): 선택 정렬, 버블 정렬 따라서 최종적으로 이진 검색 - 선형 검색 - 버블 정렬 - 선택 정렬 순이 됩니다.

Quiz 7

아래 코드는 '#'으로 피라미드를 쌓는 코드입니다. draw()와 같이 함수 안에서 함수 자기 자신을 호출하는 방식을 무엇이라고 할까요? (코드 생략)

반복(repeat)

정렬(sort)

재귀(recursive)

검색(search)

-> draw() 코드 내부에서 draw()를 다시 사용하는 이러한 방식을 재귀호출이라고 합니다. 어떤 작업을 반복적으로 해야할때 반복문 대신 활용할 수 있습니다. 재귀호출을 할때는 함수를 종료하기 위한 조건을 반드시 설정해줘야합니다.

boostcourse

Quiz 8

아래 코드와 같이 피라미드 쌓기를 재귀적으로 작성한 코드에서, h 값으로 3이 입력되었을 때 draw 함수는 총 몇 번 호출될까요? (코드 생략)

1

2

3

4

-> h 값이 3인 경우, 3일때, 2일때, 1일때 총 3번 호출되며 총 3층이 쌓이게 됩니다.

boostcourse

Quiz 9

병합 정렬, 선택 정렬, 버블 정렬의 실행시간의 하한을 빠른 순서대로 정렬한 것은 무엇인가요?

선택 정렬 - 병합 정렬 - 버블 정렬

버블 정렬 - 병합 정렬 - 선택 정렬

버블 정렬 - 선택 정렬 - 병합 정렬

병합 정렬 - 선택 정렬 - 버블 정렬

-> 각 정렬 알고리즘의 하한(최선인 경우)은 다음과 같습니다. Ω(n): 버블 정렬 < Ω(n log n): 병합 정렬 < Ω(n^2): 선택 정렬. 따라서 버블 정렬 - 병합 정렬 - 선택 정렬 순으로 빠릅니다.

Quiz 10

알고리즘의 실행 시간의 상한을 비교하기 위해 Big-O 표기법을 사용합니다. 다음 Big-O 표기법 중 빠른 순서대로 올바르게 정렬한 것은 무엇인가요?

O(log n) – O(n log n) – O(n) – O(n^2)

O(log n) – O(1) – O(n) – O(n^2)

O(1) – O(log n) – O(n) – O(n^2)

O(1) – O(n log n) – O(n) – O(n^2)

-> Big-O 표기법은 알고리즘의 실행 시간이 n에 따라 변화하는 정도를 나타낸 것입니다. n에 대한 함수를 생각했을 때 그 절대적인 크기는 1 < log n < n < nlog n < n^2 순으로 커집니다(n != 1). 따라서 실행 시간의 상한 크기도 이와 동일하게 비교할 수 있습니다.