CSE12 - Lecture 10

Monday, April 22, 2024 10:00 AM

PA3 due Wednesday @ 8am

Exam 1 - Wednesday

- https://ucsd-cse12-sp24.github.io/lectures/exam1.html

Lecture 10

Categorizing Runtimes

Let f(n) = 100

Which of the following is **NOT** a correct bound?

A. f(n) is O(2^n)
B. f(n) is O(n^2)
C. f(n) is O(n)
D. f(n) is O(n^100)
E. None of these

Let f(n) = $\frac{3n^3 + 2n + 7}{n^2}$

Which of the following is a correct bound?

A. f(n) is O(log(n))
B. f(n) is O(n^2)
C. f(n) is O(n)
D. f(n) is O(n^3)
E. None of these

```
void printAllElementOfArray(int[] arr) {
    for (int i = 0; i < arr.length; i++) {
        printf("%d\n", arr[i]);
    }
}
```

Which of the following is a correct bound?

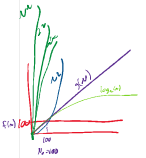
A. f(n) is O(log(n))
B. f(n) is O(n^2)
C. f(n) is O(n)
D. f(n) is O(n^3)
E. None of these

Big-O — upper bound

f(n) = O(g(n)), f(n) ____ c * g(n) for all n ≥ n0

For each function in the list below, it is related to the function below it by O, and the reverse is not true. That is, n is O(n^2) but n^2 is not O(n).

- f(n) = 1/(n^2)
- f(n) = 1/n
- f(n) = 1
- f(n) = log(n)
- f(n) = sqrt(n)
- f(n) = n
- f(n) = n^2
- f(n) = n^3
- f(n) = n^4
- ... and so on for constant polynomials ...
- f(n) = 2^n
- f(n) = n!
- f(n) = n^n



6119

Name: _____ PID: _____ Code: _____

Which of the following is a correct bound?

A. f(n) is O(log(n))
B. f(n) is O(n^2)
C. f(n) is O(n)
D. f(n) is O(n^3)
E. None of these

```
void printAllPossibleOrderedPairs(int arr[]) {
    for (int i = 0; i < arr.length; i++) {
        for (int j = 0; j < arr.length; j++) {
            printf("%d = %d\n", arr[i], arr[j]);
        }
    }
}
```

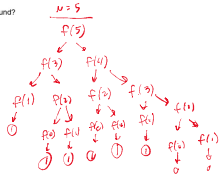
Which of the following is a correct bound?

A. f(n) is O(log(n))
B. f(n) is O(n^2)
C. f(n) is O(n)
D. f(n) is O(n^3)
E. None of these

```
int fibonacci(int num) {
    if (num <= 1) return num;
    return fibonacci(num - 2) + fibonacci(num - 1);
}
```

Which of the following is a correct bound?

A. f(n) is O(2^n)
B. f(n) is O(n^2)
C. f(n) is O(n)
D. f(n) is O(n^3)
E. None of these



Sorted

```
// A recursive binary search function. It returns location
// of x in given array arr[l..r] is present, otherwise -1
int binarySearch(int arr[], int l, int r, int x)
{
    if (r >= l) {
        int mid = l + (r - l) / 2;

        // If the element is present at the middle
        // itself
        if (arr[mid] == x)
            return mid;

        // If element is smaller than mid, then
        // it can only be present in left subarray
        if (arr[mid] > x)
            return binarySearch(arr, l, mid - 1, x);

        // Else the element can only be present
        // in right subarray
        return binarySearch(arr, mid + 1, r, x);
    }

    // We reach here when element is not
    // present in array
    return -1;
}
```

What are some correct bounds for binarySearch? What is the smallest correct bound?

O(n), O(n²), O(n³).. O(log₂(n))

```
boolean isPrimeAll(int num) {
    // Check for divisors of num
    for (int i = 2; i < num; i += 1) {
        if (num % i == 0) {
            // Any divisor other than 1 or num means num is not prime
            return false;
        }
    }
    // No other divisors found means num is prime
    return true;
}
```

What is the smallest correct bound?

```
boolean isPrimeHalf(int num) {
    // Check for divisors of num
    for (int i = 0; i < num / 2; i++) {
        if (num % i == 0) {
            // Any divisor other than 1 or num means num is not prime
            return false;
        }
    }
    // No other divisors found means num is prime
    return true;
}
```

What is the smallest correct bound?

```
void printAllItemsTwice(int arr[], int size)
{
    for (int i = 0; i < size; i++) {
        printf("%d\n", arr[i]);
    }

    for (int i = 0; i < size; i++) {
        printf("%d\n", arr[i]);
    }
}
```

What is the smallest correct bound?

```
void printFirstItemThenFirstHalfThenSayHi100Times(int arr[], int size)
{
    printf("First element of array = %d\n",arr[0]);

    for (int i = 0; i < size/2; i++) {
        printf("%d\n", arr[i]);
    }

    for (int i = 0; i < 100; i++) {
        printf("Hi\n");
    }
}
```

What is the smallest correct bound?

```
void printAllNumbersThenAllPairSums(int arr[], int size)
{
    for (int i = 0; i < size; i++) {
        printf("%d\n", arr[i]);
    }

    for (int i = 0; i < size; i++) {
        for (int j = 0; j < size; j++) {
            printf("%d\n", arr[i] + arr[j]);
        }
    }
}
```

What is the smallest correct bound?