**1. Write generalize function for Fibonacci series.**

**Ans:**

```
void fibonacci(int n) {
    int a[n];
    int i;
    a[0] = 0;
    a[1] = 1;
    printf("%d\n", a[1]);
    for (i = 2; i <= n; i++) {
        a[i] = a[i - 1] + a[i - 2];
        printf("%d\n", a[i]);
    }
}
```

| *Line* | *Cost* | *No. of times* |
|---|---|---|
| *1* | *c1* | *1* |
| *2* | *c2* | *1* |
| *3* | *c3* | *1* |
| *4* | *c4* | *1* |
| *5* | *c5* | *1* |
| *6* | *c6* | *n* |
| *7* | *c7* | *n-1* |
| **8** | **c8** | **n-1** |

TSumOflist = c1(1) + c2(1) + c3(1) + c4(1) + c5(1) + c6(n) + c7(n-1) + c8(n-1)

= c1 + c2 + c3+ c4 +c5 + nc6 + nc7 – c7 + nc8 – c8

= n(c6 + c7 + c8) + 1(c1 + c2 + c3 + c4 + c5 - c7 - c8)

= O(n)

So the time complexity of the iterative method is linear, as the loop runs from 2 to n. i.e, O(n)

2. Which of the given options provides the increasing order of asymptotic complexity of functions given below?

2^n , n^(3/2),nLogn,n^(Logn)

**Ans:**

- nlog(n) is the slowest

- 2^n is fastest

2^n < n^log(n) < n^(3/2) < nlog(n)

**3. I want to find the the book from the book record database. What should be worst case time complexity to find the particular book?**

**Ans:**

- Lets assume the books database is an array.

- to search one item from that array we can use linear search.

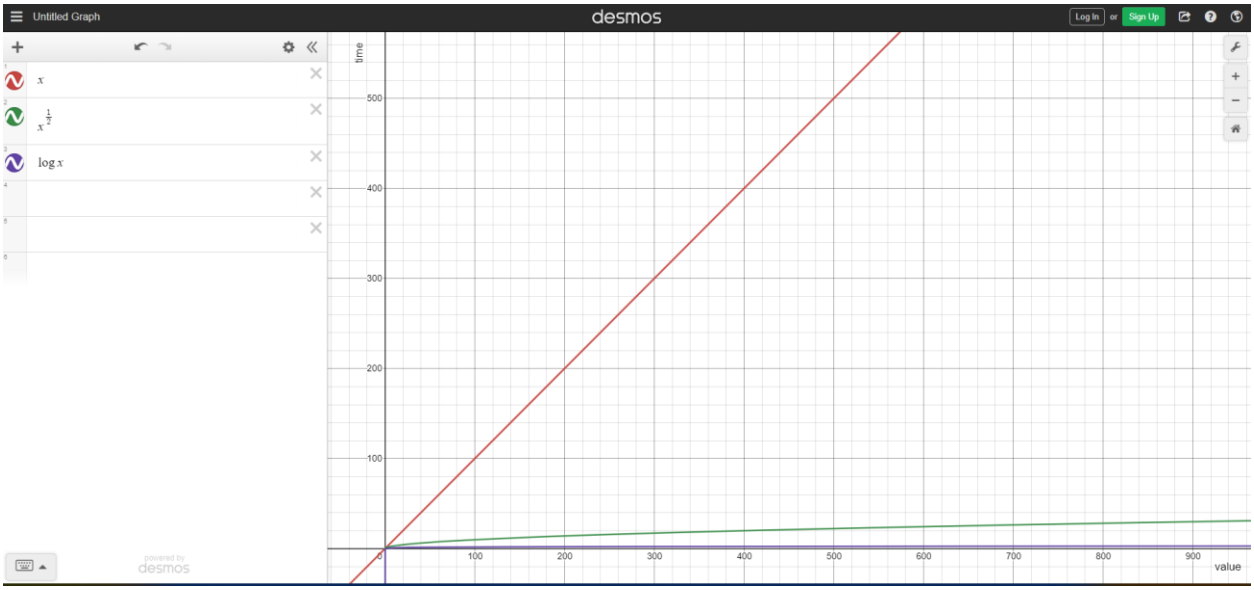
Time complexity for worst case of linear search is O(n).

**4. Time complexities of three algorithms are given. Which should execute the slowest for large values of n?**

**O(n), O(n^(1/2)), O(log(n))**

**Ans:**

As per the graph




O(n) is the slowest.