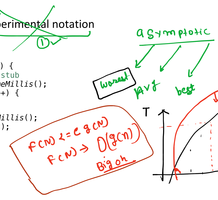
Lec-11
05/05/25 4:58 PM

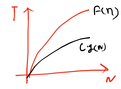
Time complexity & Space complexity

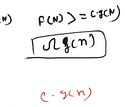
Time Complexity
asymptotic notation vs Experimental notation

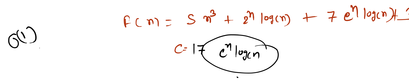
```
public static void main(String[] args) {
        // TODO Auto-generated method stub
        long start = System.currentTimeMillis();
        for (int i = 0; i < 100000; i++) {

        }
        long end = System.currentTimeMillis();
        System.out.println(end - start);
    }
```

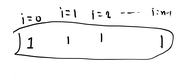




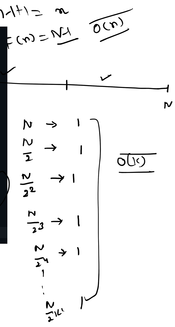


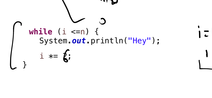



```
public static int Search(int[] arr, int item) {
    for (int i = 0; i < arr.length; i++) {
        if(arr[i]==item) {
            return i;
        }
    }
    return -1;
}
```

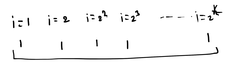


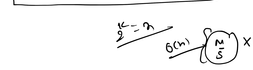

```
public static int Search(int[] arr, int item) {
    int lo = 0;
    int hi = arr.length - 1;
    while (lo <= hi) {
        int mid = (lo + hi) / 2;
        if(arr[mid]==item) {
            return mid;
        }
        else if(arr[mid]>item) {
            hi=mid-1;
        }
        else {
            lo=mid+1;
        }
    }
    return -1;
}
```



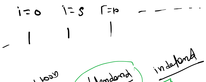



```
while (i <=n) {
    System.out.println("Hey");
    i *= 6;
}
```

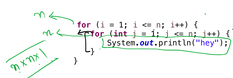


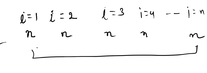



```
while (i <= n) {
    System.out.println("Hey");
    i += 2;
    i += 3;
}
```

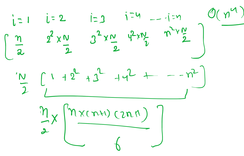



```
for (i = 1; i <= n; i++) {
    for (int j = 1; j <= n; j++) {
        System.out.println("hey");
    }
}
```

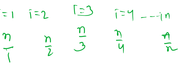




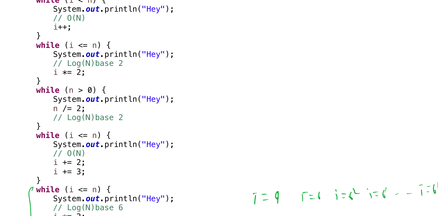

```
for (i = 1; i <= n; i++) {
    for (int j = 1; j <= i * i; j++) {
        for ( k = 1; k <= n / 2; k++) {
            System.out.println("hey");
        }
    }
}
```




```
for (i = 1; i <= n; i++) {
    for (int j = 1; j <= n; j += i) {
        System.out.println("hey");
    }
}
```




```
int i = 0;
while (i < n) {
    System.out.println("Hey");
    // O(N)
    i++;
}
while (i <= n) {
    System.out.println("Hey");
    // Log(N)base 2
    i *= 2;
}
while (n > 0) {
    System.out.println("Hey");
    n /= 2;
    // Log(N)base 2
}
while (i <= n) {
    System.out.println("Hey");
    // O(N)
    i += 2;
    i += 3;
}
while (i <= n) {
    System.out.println("Hey");
    // Log(N)base 6
    i *= 2;
    i *= 3;
}
while (n > 0) {
    System.out.println("Hey");
    // Log(N)base 6
    n /= 2;
    n /= 3;
}
int k = 2;
while (i <= n) {
    System.out.println("Hey");
    // loops->N/K
    i += k;
}
while (i <= n) {
    System.out.println("Hey");
    // Log(N)base k
    i *= k;
}
for (i = 1; i <= n; i++) {
    for (int j = 1; j <= n; j++) {
        System.out.println("hey");
        // O(N^2)
    }
}
for (i = 1; i <= n2; i++) {
    // O(Sqrt(N))
    System.out.println("hey");
}
for (i = 1; i <= n; i++) {
    for (int j = 1; j <= n * i; j++) {
        for (k = 1; k <= n / 2; k++) {
            System.out.println("hey");
            // O(N^4)
        }
    }
}
for (i = 1; i <= n; i *= 2) {
    // O(Log(N) base 2
    System.out.println("hey");
}
for (i = n / 2; i <= n; i++) {
    for (int j = 1; j <= n / 2; j++) {
        for (k = 1; k <= n; k = k * 2) {
            // O(N^2log(N))
            System.out.println("hey");
        }
    }
}
for (i = 1; i <= n; i++) {
    for (int j = 1; j <= n; j += i) {
        System.out.println("hey");
    }
}
```




```
public static int root(int n, int k) {
    int lo = 1;
    int hi = n;
    int ans = 0;
    while (lo <= hi) {
        int mid = (lo + hi) / 2;
        if (mid <= n) {
            ans = mid;
            lo = mid + 1;
        } else {
            hi = mid - 1;
        }
    }
    return ans;
}
```