DATE: __/__/____ DAY: __/__/____

Analysis of Algorithms.

Assignment 2.

Daniyal Asghar
53731

Problem 1: Linear Search.

```
int linearSearch(int arr[], int n, int target) {
    for (int i=0; i<n; i++) {
        if (arr[i] == target) {
            return i; } }
    return i; }}
```

Explanation:

-> Single loop run from 0 - $n-1
-> Each iteration perform 1 compatation.
-> warst case: all "n" elements checked
-> Best case: found at index 1,

Calculation:

$$T(n) = n$$
$$O(n)$$

P.2: Binary Search:

```
int binarySearch(int [], int left, int right, int target) {
    while (left <= right) {
        int mid = left+(right-left)/2;
        if(arr[mid]==(target) return mid;
        if(arr[mid] <target) left=mid+1;
        else right = mid-1; }
```

DATE: __/__/____ DAY: __/__/____

```
return -1;
}
```

Explanation:

→ Each interation, half will arr size
→ Max step = $\log_2 n$
→ Each step does constant coompesion
→ Best case: found at first mid.

Calculation:

interval after ith step = $n/2^k$

$n/2^k \leq 1$ ∴ $k = \log_2 n$

$T(n) = \lceil \log_2 n \rceil$

→ $O(\log n)$

P.3: Bobble Sort:

```
void bubbleSort(int arr[], int n) {
    for (int i=0; i<n-1; i++) {
        for (int j=0; j<n-i-1; j++) {
            if (arr[j] > arr(j+1) {
                std::swap(arr[j], arr(j+1));
}}}}
```

Explanation

Tube nested loops
atlar loop runs n-1 time.
Inner loop runs (n-1) → (n-2) ... +1

Calculation:

Inner loop = 1+2+ ..... → (n-1)

(n-1) n/2 = (n²-n)/2

T(n) = n²/2

$O(n^2)$

DATE: __/__/____ DAY: __/__/____

4. Selection Sort:

Explanation:
outer loop runs n-1
Inner loop compare elements to find min.
Total comparisons $\approx (n-1)+(n-2)+\cdots 1$
Swap = n-1 (negligible)

Calculation:
Total Compocision $= n-1)n/2 = (n^2-n)/2$
$T(n) = n^2/2$
$O(n^2)$

5. Matrix Multiplication.

Explanation:
Three nested loop (i, j, k)
i → n interation
j → n
k → n
Total $= n \times n \times n = O(n^3)$

DATE: __/__/____ DAY: __/__/____

6 : Fibonacci (Recursive)

Explanation.

Each call generates 2 more call
Recursion tree double at each level
Growth is exponential,

Calculation:

$T(n) = T(n-1) + T(n-2) + O(1)$
No, call $\approx 2^n$.

$O(2^n)$

7. Fibonacci (iterative)

Explanation:

One loop run from 2 to n
Each iteration dose constant work
Growth is linear.

Calculation:

loop executes $(n-1)$ time
$T(n) = n \times O(1) = n$.

$O(n)$

DATE: __/__/____ DAY: __/__/____

8. Factorial (Recursive)

Explanation.

Each cell reduce n by 1.

Total call = n+1 (including base)

Each cald close constant work.

Calculation.

Recursive: $T(n) = T(n-1) + 1$

$T(n) = n + 1$

$O(n)$

9: Nested loop

Explanation:

outer loop run n time

inner loop also run n time.

Total iteration = n x n

calculation

$n \times n = n^2$

Each iteration = constant printing

$T(n) = n^2$

$O(n^2)$

DATE: __/__/____ DAY: __/__/____

10: Recursive Power function:

Explanation:

Each call decreases n by 1
Total call = n+1 (including base
Each call does constant Multiplication
linear growth.

Calculation:

$$T(n) = T(n-1) + 1$$
$$T(n) = n + 1$$

$$O(n)$$