This code contains multiple programming problems and their solutions in C and C+

SESSION 1: Searching

* **Dreamplay likes the string... (LVL 3):** This code counts the number of step

```c
#include <stdio.h>
#include <string.h>

int count(char *s, int n) {
    int steps = 0;
    for (int i = 0, j = n - 1; i < j; i++, j--) {
        if (s[i] != s[j]) {
            steps++;
        }
    }
    return steps;
}

int main() {
    char s[5001];
    scanf("%s", s);
    int n = strlen(s);
    printf("%d\n", count(s, n));
    return 0;
}
```

* **You are given two numbers, namely A and S (LVL 3):** This code counts the nu

```c
#include <stdio.h>
#define MOD 1000000009
```

```
int countWays(int A, int S) {
    int dp[A + 1];
    for (int i = 0; i <= A; i++)
        dp[i] = 0;
    dp[0] = 1;
    for (int i = S; i <= A; i++) {
        for (int j = i; j <= A; j++) {
            dp[j] = (dp[j] + dp[j - i]) % MOD;
        }
    }
    return dp[A];
}

int main() {
    int t, A, S;
    scanf("%d", &t);
    while (t--) {
        scanf("%d %d", &A, &S);
        printf("%d\n", countWays(A, S));
    }
    return 0;
}
```

* **Aliens and Predators (LVL 2):** This code uses Depth-First Search (DFS) to d

* **Wef and Astro (LVL 3):** This code counts the number of unique strings (afte

SESSION 2: Sorting Techniques

* **Lets Consider some weird country (LVL 3):** This is a complex graph problem

* **Karter wants to celebrate (LVL 2):** This code finds the maximum sum of valu

* **All Road of wonderland land (LVL 3):** This code finds the minimum number of

* **Monk is given a tree rooted at Node (LVL 3):** This code calculates the prod

* **a permutation is a list (LVL 3):** This code processes a permutation and a g

SESSION 3: Divide and Conquer

* **A newspaper is published in wonderland (LVL 3):** This code counts the minim

* **Rajesh has given an array a (LVL 2):** This code aims to find the minimum ab

* **Victor Valmiki and Justin Array (LVL 3):** This code calculates the reduced

General Improvements

* **Error Handling:** Many of these codes lack proper error handling (e.g., chec
* **Memory Management:** Several solutions use `malloc` without corresponding `
* **Efficiency:** Some solutions have nested loops that lead to O(n^2) or worse
* **Code Style:** Consistency in code style (e.g., indentation, naming conventi
* **Comments:** Add more comments to explain complex logic and algorithms.

Remember that for larger problems, you should always consider the time and space