13. Write C program that demonstrates the usage of these notations by analyzing the time complexity of some example algorithms.

Code:

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
void constantTime(int n) {
    int x = n * n;
    printf("Constant Time Operation: %d\n", x);
}
void logarithmicTime(int n) {
    while (n > 1) {
        n = n / 2;
        printf("Logarithmic Step\n");
    }
}
void linearTime(int n) {
    for (int i = 0; i < n; i++) {
        printf("Linear Step: %d\n", i);
    }
}
void linearithmicTime(int n) {
    for (int i = 0; i < n; i++) {
        int j = i;
        while (j > 1) {
            j = j / 2;
            printf("Linearithmic Step: %d\n", j);
        }
    }
}
```

```c
void quadraticTime(int n) {
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            printf("Quadratic Step: (%d, %d)\n", i, j);
        }
    }
}

void cubicTime(int n) {
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            for (int k = 0; k < n; k++) {
                printf("Cubic Step: (%d, %d, %d)\n", i, j, k);
            }
        }
    }
}
int main() {
    int n = 10;
    printf("Analyzing Time Complexity with n = %d\n", n);
    printf("\nConstant Time:\n");
    constantTime(n);
    printf("\nLogarithmic Time:\n");
    logarithmicTime(n);
    printf("\nLinear Time:\n");
    linearTime(n);
    printf("\nLinearithmic Time:\n");
    linearithmicTime(n);
    printf("\nQuadratic Time:\n");
    quadraticTime(n);
    printf("\nCubic Time:\n");
```

```c
    cubicTime(n);
    return 0;
}
```

Output:

```
Analyzing Time Complexity with n = 10

Constant Time:
Constant Time Operation: 100

Logarithmic Time:
Logarithmic Step
Logarithmic Step
Logarithmic Step

Linear Time:
Linear Step: 0
Linear Step: 1
Linear Step: 2
Linear Step: 3
Linear Step: 4
Linear Step: 5
Linear Step: 6
Linear Step: 7
Linear Step: 8
Linear Step: 9

Linearithmic Time:
Linearithmic Step: 1
Linearithmic Step: 1
Linearithmic Step: 2
Linearithmic Step: 1
Linearithmic Step: 2
Linearithmic Step: 1
```

Time Complexity:

- T(n)= O(n)