Experiment 2

StudentName: Rohit Kumar	**UID: 22BCS15717**
Branch:BE-CSE	**Section/Group: FL_IOT_603‘B**
Semester: 5th	**Date of Performance:25/07/24**

Subject Name:Design and Analysis Algorithms
Subject Code:22CSH-311

1. **Aim:** Write a code to implement power function in O(logn) time complexity

2. **Objective:** To implement power function in O(logn) time complexity.

3. **Algorithm**

Start.
Input: Prompt the user to enter the base and the exponent.
Read: Read the base value.
Read: Read the exponent value.
Calculate Power: Call `calculatePower(base, exponent)`.

- **Base Case**:
  - If `exponent == 0`, return 1.
- **Recursive Case**:
  - Compute `halfPower = calculatePower(base, exponent/2)`.
- **Even Case**:
  - If `exponent % 2 == 0`, return `halfPower * halfPower`.
- **Odd Case**:
  - If `exponent % 2 != 0`, return `base * halfPower * halfPower`.

Output: Display the result of `calculatePower(base, exponent)`.
End.

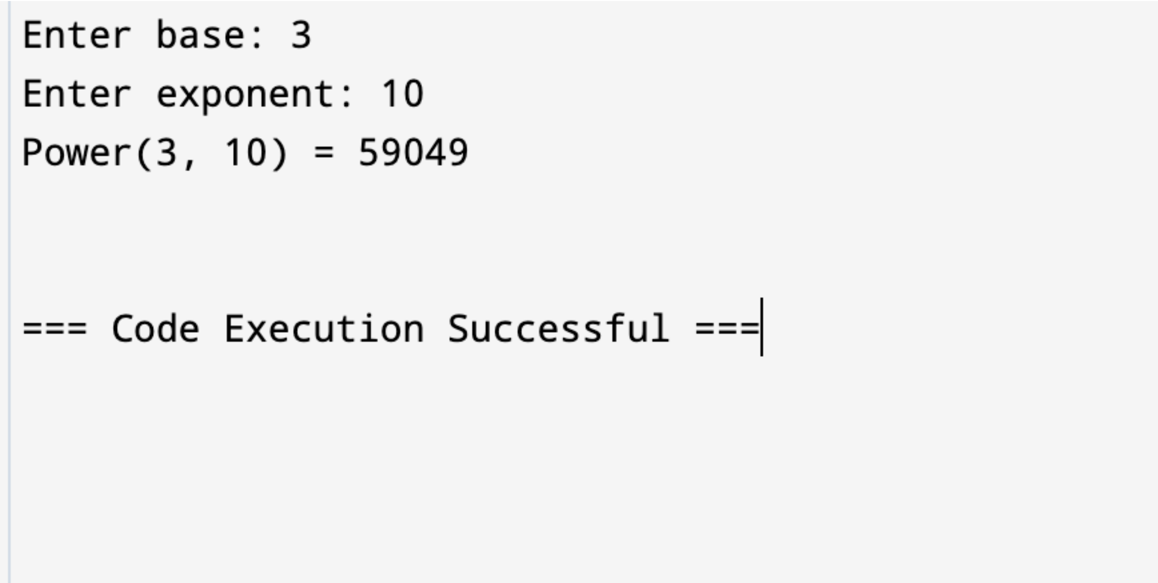
4. **Implementation/Code:**

```
#include <iostream>

using namespace std;
```

```
int calculatePower(int base, int exponent) {
    if (exponent == 0)
        return 1;
    int halfPower = calculatePower(base, exponent / 2);
    if (exponent % 2 == 0)
        return halfPower * halfPower;
    else
        return base * halfPower * halfPower; }
int main() {
    int base, exponent;
    cout << "Enter base: ";
    cin >> base;
    cout << "Enter exponent: ";
    cin >> exponent;
    cout << "Power(" << base << ", " << exponent << ") = " <<
calculatePower(base, exponent) << endl;
    return 0;
}
```

5. Output



```
Enter base: 3
Enter exponent: 10
Power(3, 10) = 59049


=== Code Execution Successful ===
```

6. Time Complexity

The time complexity is O(log n).

7. Learning Outcome:

1) Learnt how to compute large powers efficiently, improving from a O(n)approach to O(log n).
2) 2) Learnt implementing recursive algorithms.