Marwadi University Marwadi Chandarana Group	**Marwadi University** **Faculty of Engineering and Technology** **Department of Information and Communication Technology**	
Subject: Design and Analysis of Algorithm	**Aim: Implementing the Exponential (Power) function to calculate x^n using Divide and Conquer Approach**	
Experiment No: 03	**Date: 12/08/2025**	**Enrollment No: 92301733049**

LAB EXPERIMENT - 3
NAME: Shivani Ambati
ROLL NO: 92301733049
DATE: 12/08/25

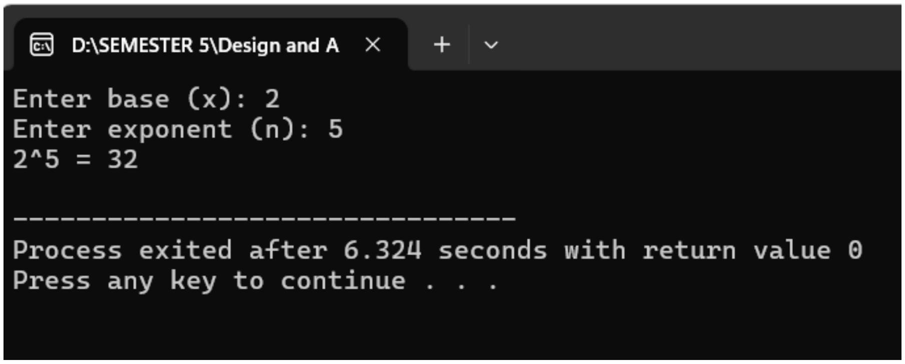
AIM: To implement the power function (x^n) using different approaches
METHODS:

1. Simple loop method (iterative)
2. Recursive method (divide and conquer but O(N))
3. Optimized recursive method (divide and conquer O(logN))

```
#include <iostream>
using namespace std;
```

Method 1: Simple iterative approach - O(N)

```
// Just multiply x n times
double power_iterative(double x, int n) {
    double ans = 1.0;
    for(int i = 0; i < n; i++) {
        ans = ans * x;  // keep multiplying
    }
    return ans;
}
```

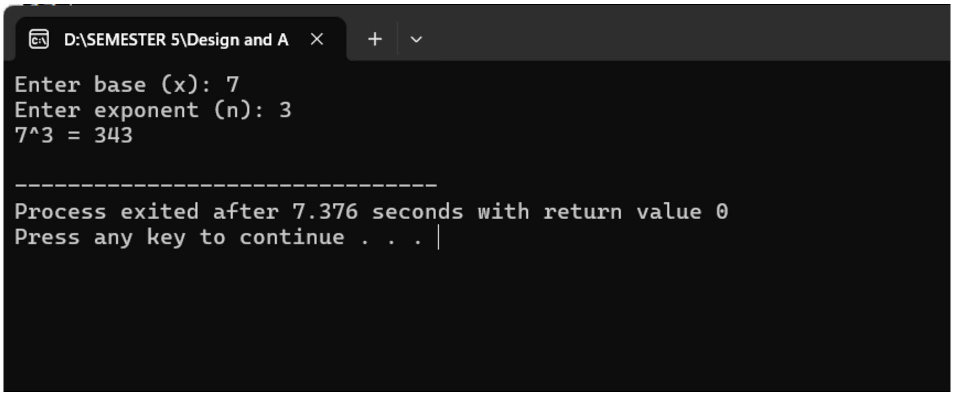


```
Enter base (x): 2
Enter exponent (n): 5
2^5 = 32

--------------------------------
Process exited after 6.324 seconds with return value 0
Press any key to continue . . .
```

Method 2: Recursive approach - O(N)

```
// Breaking problem into smaller subproblems
double power_recursive(double x, int n) {
    // base case
    if(n == 0) {
        return 1;
    }
    // recursive case
    return x * power_recursive(x, n-1);
}
```

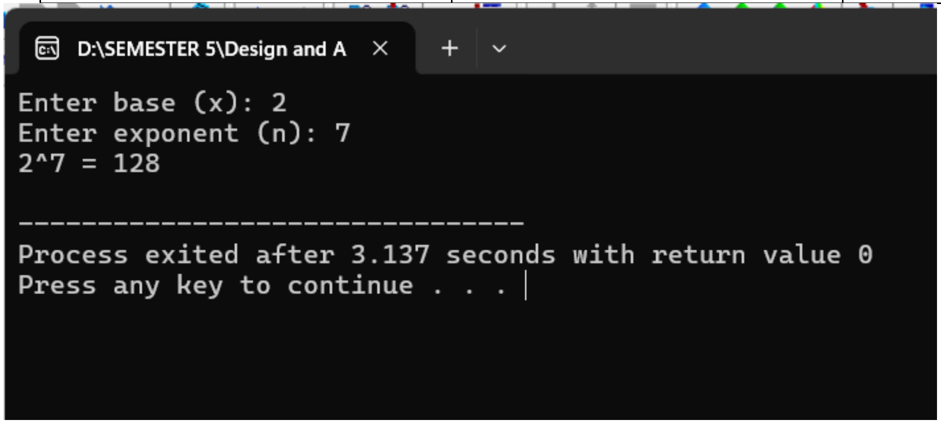


```
Enter base (x): 7
Enter exponent (n): 3
7^3 = 343

--------------------------------
Process exited after 7.376 seconds with return value 0
Press any key to continue . . . 
```

Method 3: Optimized recursive approach - O(logN)

```
// Using exponentiation by squaring
double power_optimized(double x, int n) {
    if(n == 0) {
        return 1;
    }
    // if n is even
    if(n % 2 == 0) {
        double temp = power_optimized(x, n/2);
        return temp * temp;
    }
    // if n is odd
    else {
        return x * power_optimized(x, n-1);
    }
}
```



```
Enter base (x): 2
Enter exponent (n): 7
2^7 = 128

--------------------------------
Process exited after 3.137 seconds with return value 0
Press any key to continue . . .
```

```
int main() {
   double x;
   int n;

   cout << "This program calculates x^n using different methods\n";
   cout << "--------------------------------------------------\n";

   // Taking input from user
   cout << "Enter the base (x): ";
   cin >> x;
   cout << "Enter the exponent (n): ";
   cin >> n;

   cout << "\nRESULTS:\n";
   cout << "1. Using simple iterative method: " << power_iterative(x, n) << endl;
   cout << "2. Using recursive method: " << power_recursive(x, n) << endl;
   cout << "3. Using optimized recursive method: " << power_optimized(x, n) << endl;

   cout << "\nAll methods give same result when correct!\n";
   return 0;
}
```