NATIONAL INSTITUTE OF TECHNOLOGY RAIPUR

B.Tech. (5th Semester)

Assignment No :- 2

Department of Computer Science & Engineering

Subject: Advance data Structure

Lab Code- CS105201CS

Submitted By :- Jayesh Sharma

Roll No :- 23115045

Lab Batch No :- 1

Q.1) Related Algorithms with their Best and Average Case complexities :-

Efficiency	Big-O	Iterations	Estimated Time	Related Algorithms	Best Case	Average Case
Logarithmic	O(log n)	14	microseconds	Binary Search, AVL Trees	O(1)	O(log n)
Linear	O(n)	10,000	seconds	Linear Search,	O(1)	O(n)
Linear Logarithmic	O(n log n)	140,000	seconds	Heap Sort , Merge Sort	O(n log n)	O(n log n)
Quadratic	$O(n^2)$	$10,000^2$	minutes	Bubble Sort, Insertion Sort	O(n)	$O(n^2)$
Polynomial (n^k)	$O(n^3)$	10^{12}	hours	Matrix Multiplication	$O(n^2)$	$O(n^3)$
Exponential	$O(c^n)$	2^10,000	Intractable	Recursive Fibonacci	O(1)	$O(2^n)$
Factorial	O(n!)	10,000!	Intractable	Traveling Salesman (Brute)	$O(n^2)$ (approx)	O(n!)

Q.2) Related Algorithms with their Best and Average Case complexities :-

A) . Fibonacci Number :-

```python
def fibonacci(n):
    if n <= 1:
        return n
    return fibonacci(n - 1) + fibonacci(n - 2)
```

B) . Greatest Common Divisor:-

```python
def gcd(a, b):
    if b == 0:
        return a
    return gcd(b, a % b)
```

C) . Power Function:-

```
def power(a, b):
    if b == 0:
        return 1
    return a * power(a, b - 1)
```

D) . Sum of Digits:-

```
def sum_digits(n):
    if n == 0:
        return 0
    return (n % 10) + sum_digits(n // 10)
```

E) . Check if array Sorted :-

```
def is_sorted(arr, n):
    if n == 0 or n == 1:
        return True
    if arr[n - 1] < arr[n - 2]:
        return False
    return is_sorted(arr, n - 1)
```

Q.3) Develop a program using a recursive algorithm that changes an integer to a binary number :-

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

void toBinary(int n) {
    if (n == 0) return;
    toBinary(n / 2);
    cout << n % 2 <<endl;
}

int main() {
    int num; cin >> num;
    if (num == 0) cout << 0<<endl;
    else toBinary(num);
    return 0;
}
```

Q.4) Write a program to implement/represent stack and its operations (push and pop) using linked list :-

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

struct Node {
    int data;
    Node* next;
};

class Stack {
    Node* top;
public:
    Stack() { top = nullptr; }

    void push(int val) {
        Node* temp = new Node;
        temp->data = val;
        temp->next = top;
        top = temp;
    }

    void pop() {
        if (top == nullptr) return;
        Node* temp = top;
        top = top->next;
        delete temp;
    }

    void display() {
        Node* temp = top;
        while (temp != nullptr) {
            cout << temp->data << " -> ";
            temp = temp->next;
        }
        cout << "NULL" << endl;
    }
};

int main() {
    Stack s;
    s.push(10); s.push(20); s.push(30);
    s.display(); s.pop(); s.display();
    return 0;
}
```

Q.5) Write a program to implement a simple STACK application program. The data is random uppercase characters. After the characters have been inserted they are popped and printed. When the stack is empty the program gets terminated :-

```
#include <iostream>
#include <stack>
#include <cstdlib>
#include <ctime>
using namespace std;

int main() {
    stack<char> s;
    srand(time(0));
    for (int i = 0; i < 10; i++) {
        char ch = 'A' + rand() % 26;
        cout << "Pushed: " << ch << endl;
        s.push(ch);
    }

    cout << "\nPopping from stack:\n";
    while (!s.empty()) {
        cout << s.top() << endl;
        s.pop();
    }
    return 0;
}
```