RECURSION - 2 | Week 10

1. **CALCULATE SUM OF DIGITS USING RECURSION.**

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

// Function to calculate sum of digits recursively
int sum_of_digits(int n) {
    // Base case: if n is a single digit
    if (n < 10) {
        return n;
    } else {
        // Recursive case: add the last digit to the sum of the digits of the remaining
number
        return n % 10 + sum_of_digits(n / 10);
    }
}

int main() {
    int number = 12345;
    int result = sum_of_digits(number);
    cout << "The sum of the digits of " << number << " is " << result << endl;

    return 0;
}
```

2.CALCULATE REVERSE OF A NUMBER USING RECURSION.

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

// Function to calculate reverse of a number recursively
int reverse_number(int n, int rev = 0) {
    // Base case: if n becomes 0, return the reversed number
    if (n == 0) {
        return rev;
    } else {
        // Extract the last digit of n
        int last_digit = n % 10;
        // Append the last digit to rev (reversed number)
        rev = rev * 10 + last_digit;
        // Recursive call to process remaining digits
        return reverse_number(n / 10, rev);
    }
}

int main() {
    int number = 12345;
    int reversed = reverse_number(number);
    cout << "The reverse of " << number << " is " << reversed << endl;

    return 0;
}
```

3.
NUMBERS OF STEPS TO REDUCE A NUMBER ZERO.

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

// Function to calculate the minimum number of steps to reduce a number to zero
int reduce_to_zero(int n) {
    // Base case: if n is already zero, no steps are needed
    if (n == 0) {
        return 0;
```

```
    }

    // Recursive cases:
    if (n % 2 == 0) {
        // If n is even, divide it by 2
        return 1 + reduce_to_zero(n / 2);
    } else {
        // If n is odd, subtract 1 from it
        return 1 + reduce_to_zero(n - 1);
    }
}

int main() {
    int number = 15;  // Example number to reduce to zero
    int steps = reduce_to_zero(number);
    cout << "Minimum steps to reduce " << number << " to zero: " << steps << endl;

    return 0;
}
```

4.PREDICT THE OUTPUT

```
INT FUN(N){
IF(N<=1) RETURN 1;
IF(N%2==0) RETURN FUN(N/2);
RETURN FUN(N/2)+FUN(N/2+1);
}
```

OUTPUT:-

```
fun(5) = 3
fun(8) = 1
fun(10) = 3
```