Assignment Solutions | Recursion - 1 | Week 11

1. Write a program to calculate the sum of odd numbers between a and b (both inclusive) using recursion.

Solution:

```
#include <bits/stdc++.h>
using namespace std;
int findSum(int curr, int lastNumber) {
    if(curr > lastNumber) return 0;
    if(curr % 2 == 0) return findSum(curr+1, lastNumber);
    return curr + findSum(curr+2, lastNumber);
}
int main() {
    int a, b;
    cin >> a >> b;
    cout << findSum(a, b) << endl;
    return 0;
}
```

2. Calculate the number of ways in which a person can climb n stairs if he can take exactly 1, 2 or 3 steps at each level.

Solution:

```
#include <bits/stdc++.h>
using namespace std;
int findNumberOfWays(int n) {
    if(n < 0) return 0;
    if(n == 0)return 1;
    return findNumberOfWays(n-1) + findNumberOfWays(n-2) + findNumberOfWays(n-3);
}
int main() {
    int n;
    cin >> n;
    cout << findNumberOfWays(n) << endl;
    return 0;
}
```

3. Given a positive integer, return true if it is a power of 2.

Solution:

```
#include <bits/stdc++.h>
using namespace std;
bool isPowerOfTwo(int n) {
    if(n == 1) {
        return true;
    }
    if(n % 2 == 0) {
        return isPowerOfTwo(n / 2);
    }
    return false;
}

int main() {

    int n;
    cin >> n;

    if(isPowerOfTwo(n)) {
        cout << "Yes" << endl;
    } else {
        cout << "No" << endl;
    }
```
