Group H.W.

5. a) $(0,2,4,6)$

$(0,3,6)$

$(0,4,6)$

b) Let $f(n)$ = number of ways to get to pad n

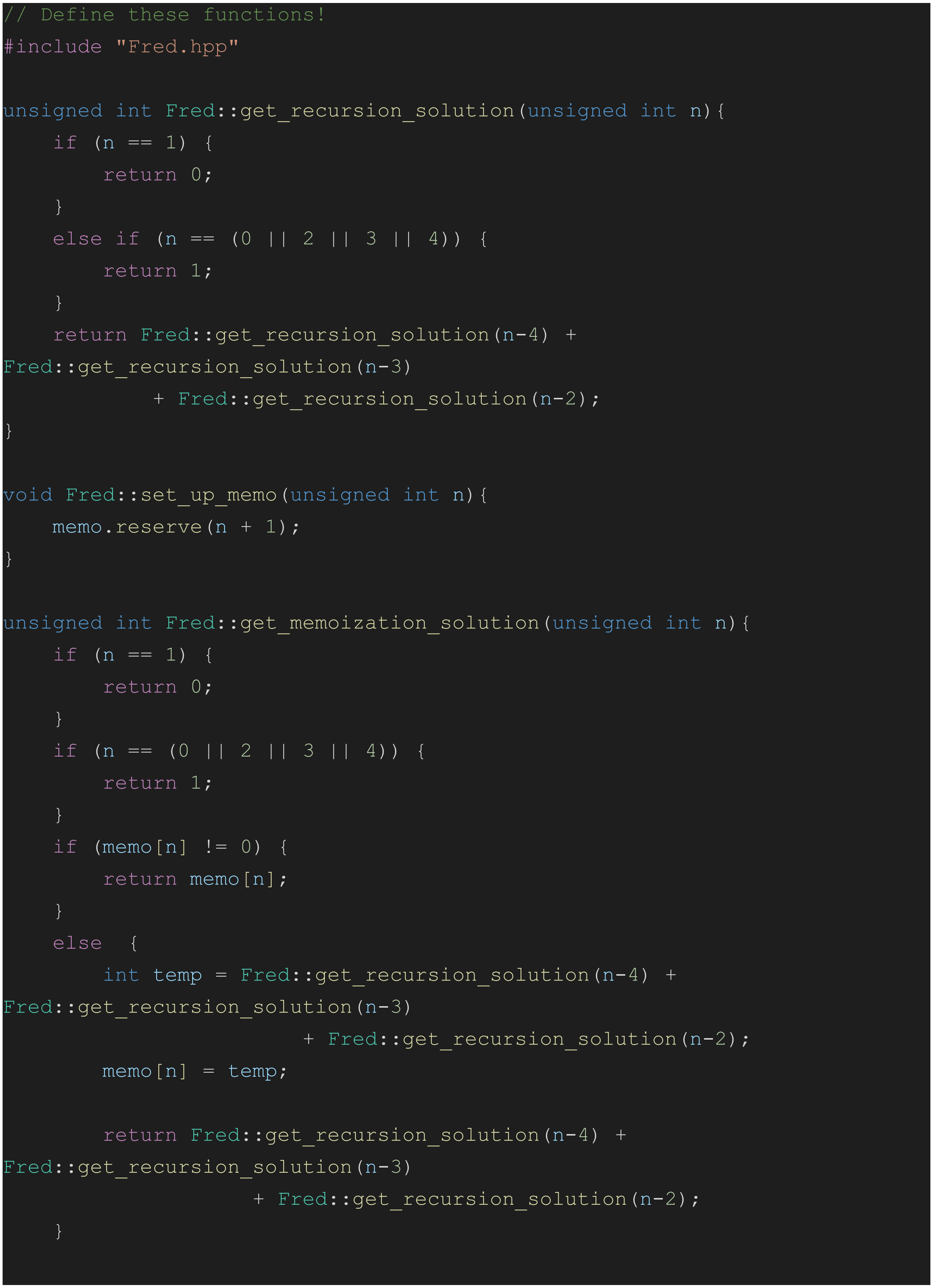
$$f(n) = f(n-4) + f(n-3) + f(n-2) \quad \forall n \geq 5$$

Base cases:

$f(0) = 1$ $\quad f(2) = 1$ $\quad f(4) = \underline{\underline{1}}$ way

$f(1) = \underline{0}$ ways $\quad f(3) = 1$

$$f(6) = f(6-2) + f(6-3) + f(6-4)$$
$$= f(4) + f(3) + f(2)$$
$$= 1 + 1 + 1$$
$$= \underline{\underline{3}} \text{ ways}$$



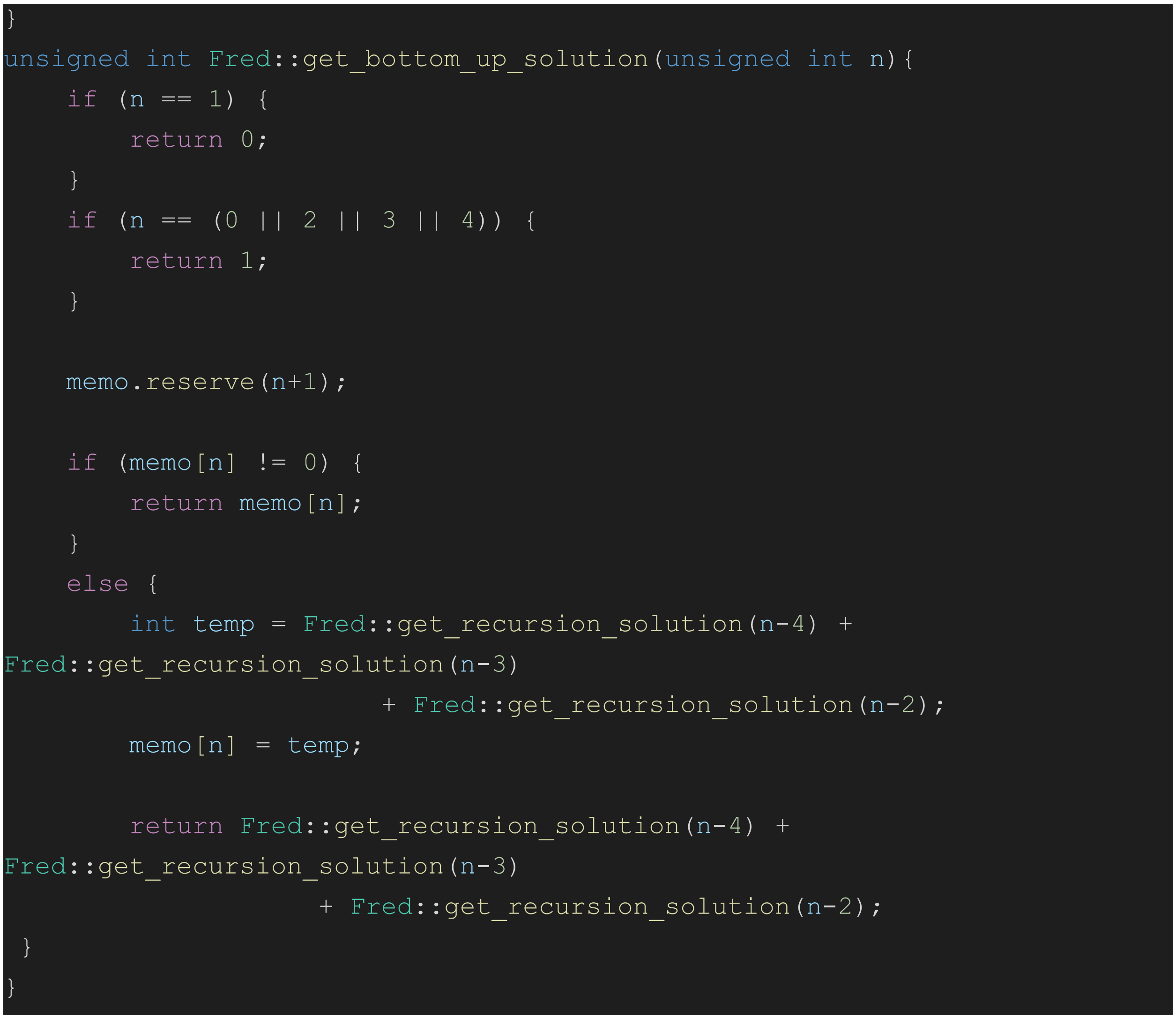
```cpp
// Define these functions!
#include "Fred.hpp"

unsigned int Fred::get_recursion_solution(unsigned int n){
    if (n == 1) {
        return 0;
    }
    else if (n == (0 || 2 || 3 || 4)) {
        return 1;
    }
    return Fred::get_recursion_solution(n-4) +
Fred::get_recursion_solution(n-3)
            + Fred::get_recursion_solution(n-2);
}

void Fred::set_up_memo(unsigned int n){
    memo.reserve(n + 1);
}

unsigned int Fred::get_memoization_solution(unsigned int n){
    if (n == 1) {
        return 0;
    }
    if (n == (0 || 2 || 3 || 4)) {
        return 1;
    }
    if (memo[n] != 0) {
        return memo[n];
    }
    else  {
        int temp = Fred::get_recursion_solution(n-4) +
Fred::get_recursion_solution(n-3)
                        + Fred::get_recursion_solution(n-2);
        memo[n] = temp;

        return Fred::get_recursion_solution(n-4) +
Fred::get_recursion_solution(n-3)
                    + Fred::get_recursion_solution(n-2);
    }
```



```
}
unsigned int Fred::get_bottom_up_solution(unsigned int n){
    if (n == 1) {
        return 0;
    }
    if (n == (0 || 2 || 3 || 4)) {
        return 1;
    }

    memo.reserve(n+1);

    if (memo[n] != 0) {
        return memo[n];
    }
    else {
        int temp = Fred::get_recursion_solution(n-4) +
Fred::get_recursion_solution(n-3)
                        + Fred::get_recursion_solution(n-2);
        memo[n] = temp;

        return Fred::get_recursion_solution(n-4) +
Fred::get_recursion_solution(n-3)
                    + Fred::get_recursion_solution(n-2);
 }
}
```