```
def F(n):
    #base case
    if n <= 1:
        return n
    #recursive case
    else:
        return (F(n-1) + F(n-2))

def f(n):
    n1 = 0
    n2 = 1
    #case 1
    if n < 0:
        return 'Incorrect input'
    #case 2
    elif n == 0:
        return n1
    #case 3
    elif n == 1:
        return n2
    #implementation of sequence
    else:
        for i in range(1, n):
            nth = n1 + n2
            n1 = n2
            n2 = nth
        return n2
```

c) $F(5)$

$F(4) + F(3)$

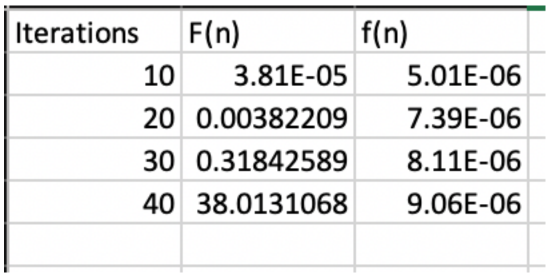
$F(3) + F(2) + F(2) + F(1)$

$F(2) + F(1) + F(1) + F(0) + F(1) + F(0) + 1$

$F(1) + F(0) + 1 + 0 + 1 + 0 + 1$

$1 + 0 + 1 + 1 + 1 + 0 + 1$

5



Iterations	F(n)	f(n)
10	3.81E-05	5.01E-06
20	0.00382209	7.39E-06
30	0.31842589	8.11E-06
40	38.0131068	9.06E-06

From the numbers in the table, it is clear that the recursive function takes much longer to run that the iterative solution. As the nth value increases, the recursive function takes much longer to run. This is due to the fact that a recursive function has more function calls than the iterative solution, so it will generally take longer.