Implement the Fibonacci Sequence using Dynamic Programming

In [1]:
```
import time
calcs = 0
def fibonacci(num):
    global calcs
    calcs += 1
    if num < 2:
        return num
    return fibonacci(num - 1) + fibonacci (num - 2)
```

In [2]:
```
# With Dynamic Programming

cache = dict()
calculations = 0
def dynamic_fib(num):
    global calculations
    calculations += 1
    if num in cache:
        return cache[num]
    else:
        if num < 2:
            return num
        else:
            cache[num] = dynamic_fib(num - 1) + dynamic_fib(num - 2)
            return cache[num]
```

In [3]:
```
# Testing implementation and comparing times

t1 = time.time()
print('Output:', fibonacci(30))
t2 = time.time()
total_time = t2-t1
print(f'Total time of operation for : {total_time}\nTotal of calculations:
```

```
Output: 832040
Total time of operation for : 0.39928603172302246
Total of calculations: 2692537
```

In [4]:
```
t1 = time.time()
print('Output:', dynamic_fib(30))
t2 = time.time()
total_time = t2-t1
print(f'Total time of operation: {total_time}\nTotal of calculations: {cal
```

```
Output: 832040
Total time of operation: 0.0006668567657470703
Total of calculations: 59
```