How to avoid repeated work?

```
int fib(int n) {
    if (n < 2) return n;
    else return fib(n-1) + fib(n-2);
}
```

```
int fib(int n) {
    // memoization
    if (n < 2) return n;
    else if (n in table) return value from table
    else
```
int result = \( f_6(n-1) + f_6(n-2) \)
put \((n, \text{result})\) in \( \text{table} \)
return result

```

int fib (int n)
{
    int table[n+1];
    table[0] = 0;
    table[1] = 1;
    for (int i = 2; i <= n; i++)
        table[i] = table[i-1] + table[i-2];
    return table[n];
}
```

dynamic programming

\begin{array}{|c|c|}
\hline
n & f_6(n) \\
\hline
2 & 1 \\
3 & 2 \\
4 & 3 \\
5 & 5 \\
\hline
\end{array}
factory scheduling: need each \( f(i,j) \); use DP

\[
\begin{array}{cccc}
  f & 1 & 2 & 3 & 4 \\
  0 & 6 & 8 & 13 & 18 \\
  1 & 2 & 14 & 14 & 17 \\
\end{array}
\]

DP in general:
1) Fill in values from base case of recurrence

\( O(n^2) \) time

2) Iterate through other entries in order such that when we get to an entry, have filled in entries it depends on

\[ f(0, 4) + x[0] > f(1, 4) + x[1], \text{ so} \]

should exit line 1

what were last few steps?