CS 310
Assignment 1023
22 October 2020

Bill Jin

Problem 1 — The nth Fibonacci number can be computed by dynamic programming. Since this is not an optimization problem, we will name the function `val` rather than the usual `opt`. Step 2 of the dynamic programming methodology results in the definition

$$\text{val}(i) = \begin{cases} 1 & \text{if } i < 2 \\ \text{val}(i-1) + \text{val}(i-2) & \text{if } i \geq 2 \end{cases}$$

Show the code that would result from implementing this definition in memoized C++ code as would be done in step 3 of dynamic programming. Show *only* the function `val`, not the main program or any helper code.

Answer: Implementing the above definition as a recursive, memoized C++ function is straightforward. We have a 1-dimension table which assigns all elements with max number it could have, which are used to show if the cell has or has not been calculated.

```
unsigned val(unsigned value, vector<unsigned>& memo)
{
  if (memo.at(value) == UINT_MAX)
  {
    if (value < 2)
    {
      memo.at(value) = 1;
    }
    else if (value >= 2)
    {
      memo.at(value) = val(value - 1, memo) + val(value - 2, memo);
    }
  }
  return memo.at(value);
}
```

Problem 2 — Show the filled-in memo table the code in the previous problem would create when computing the value for $n = 8$.

Answer: Since the program calculate the 8th Fibonacci number(count the very first Fibonacci number as 0th),which is also the index of 8 in memo table. All the rest of the cells are not calculated and remain the UINT_MAX.

1	1	2	3	5	8	13	21	32	UINT_MAX	UINT_MAX	UINT_MAX	..

Problem 3 — Show the filled-in memo table for making 12¢ using the denominations 1¢, 4¢, and 10¢. Use bold entries to show the traceback path of cells for the optimal set of coins used.

Answer: Since there could be two different ways to trace up those coins, therefore there are two solutions for this problem. The first one is one 10¢, two 1¢, the other is three 4¢:

The trace back is 10¢, and two 1¢:

		a												
		0	1	2	3	4	5	6	7	8	9	10	11	12
	0	**0**	**1**	**2**	3	4	5	6	7	8	9	10	11	12
i	1	0	1	**2**	3	1	2	3	4	2	3	4	5	3
	2	0	1	**2**	3	1	2	3	4	2	3	1	2	**3**

Problem 4 — Show the filled-in memo table for finding a longest common subsequence of the strings `SLWOVNNDK` and `ALWGQVNBKB`. Use bold entries to show the traceback path of cells for the LCS.

Answer: I highlight the both the head and tail of the arrow:

	-	S	L	W	O	V	N	N	D	K
-	**0**	0	0	0	0	0	0	0	0	0
A	0	**0**	0	0	0	0	0	0	0	0
L	0	0	**1**	1	1	1	1	1	1	1
W	0	0	1	**2**	2	2	2	2	2	2
G	0	0	1	2	2	2	2	2	2	2
Q	0	0	1	2	**2**	2	2	2	2	2
V	0	0	1	2	2	**3**	3	3	3	3
N	0	0	1	2	2	3	**4**	4	4	4
B	0	0	1	2	2	3	4	4	**4**	4
K	0	0	1	2	2	3	4	4	4	**5**
B	0	0	1	2	2	3	4	4	4	5