Analysis of Algorithms

Dynamic Programming

If more than one question appears correct, choose the more specific answer, unless otherwise instructed.

Concept: memoization
Assume zero-based indexing.

1. Consider memoizing this function:
   
   ```
   function f(x)
   {
     if (n == 0)
       return 0;
     else if (n == 1)
       return 1;
     else
       return f(x-1) + fib(x-2);
   }
   ```

   T or F: Even though there is only one formal parameter, you will need a two dimensional memoization table because you call recursively call the function twice.

2. T or F: With zero-based indexing, the size of a table dimension is always one more than the largest value used to index into that dimension.

3. T or F: Continuing with the previous question, the memoization table’s size has to be n.

4. T or F: Continuing with the previous question, the memoization table’s size has to be n × n.

5. T or F: If the formal parameter is getting smaller in the recursive calls, the table will be filled out from smaller to larger indices.

6. T or F: If the formal parameter is getting larger in the recursive calls, the table will be filled out from smaller to larger indices.

7. T or F: The number of formal parameters in the function is equal to the dimension of the dynamic programming table.

8. T or F: It is necessary to initialize a dynamic programming table with base case values.

9. T or F: In converting a recursive function to a dynamic programming solution, base cases can be moved to a table lookup function.

10. Consider memoizing this function:
    
    ```
    function f(x)
    {
      if (x == 0) return 0;
      if (x == 1) return 1;
      return f(x-2) + f(x-1);
    }
    ```

    What would be memoization table’s largest index/indices?

    (A) x - 2  
    (B) x + 1  
    (C) x 
    (D) x + 1 and x 
    (E) x - 1  
    (F) x + 1 and x + 1  
    (G) x and x 
    (H) x + 1 and x - 1
11. Consider memoizing this function:

```javascript
function f(x)
{
    if (x == 0) return 0;
    if (x == 1) return 1;
    return f(x-2) + f(x-1);
}
```

In order to remove the two original base cases, how would the memoization table be initialized? Assume the memoization table is named `memo`.

(A) memo[0] = 0; memo[1] = 1;  
(B) memo[1] = 1;  
(C) memo[0] = 0;  
(D) memo[0] = 1; memo[1] = 1;

12. Consider memoizing this function:

```javascript
function f(x)
{
    if (x == 0) return 0;
    if (x == 1) return 1;
    return f(x-2) + f(x-1);
}
```

What base case would replace the two original base cases, assuming a properly initialized memoization table (named `memo`)?

(A) if (memo[x] != EMPTY) return x;  
(B) if (memo[x] != EMPTY) return 0;  
(C) if (memo[x] != EMPTY) return 1;  
(D) if (memo[x] != EMPTY) return memo[x];

13. Consider memoizing this function:

```javascript
function g(x,items,y)
{
    if (x == 0) return 1; //first base case
    if (x < 0) return 0; //second base case
    if (y == items.size) return 0; //third base case
    return minimum(g(x-items[y],items,y),g(x,items,y+1));
}
```

Assuming all three original base cases are retained and the smallest memoization table possible, What would be the memoization table’s largest index/indices, where x refers to the original value of x?

(A) x - 2  
(B) x - 1  
(C) x and items.size + 1  
(D) x + 1  
(E) x  
(F) x and items.size  
(G) x and items.size - 1  
(H) x + 1 and items.size + 1

14. Consider memoizing this function:

```javascript
function g(x,items,y)
{
    if (x == 0) return 1; //first base case
    if (x < 0) return 0; //second base case
    if (y == items.size) return 0; //third base case
    return minimum(g(x-items[y],items,y),g(x,items,y+1));
}
```

Which of the original base cases cannot be removed, given no knowledge of the values in items?

(A) the first and third  
(B) the second and third  
(C) the third  
(D) the second  
(E) the first and second  
(F) the first
15. Consider memoizing this function:

```javascript
function g(x, items, y) {
    if (x == 0) return 1; // first base case
    if (x < 0) return 0; // second base case
    if (y == items.size) return 0; // third base case
    return minimum(g(x-items[y], items, y), g(x, items, y+1));
}
```

Removing all possible base cases, what would be the memoization table’s largest index/indices? Assume no knowledge of the values in items.

(A) $x$
(B) $x - 2$
(C) $x + 1$ and $\text{items.size}$
(D) $x$ and $\text{items.size} - 1$
(E) $x$ and $\text{items.size}$
(F) $x - 1$
(G) $x + 1$ and $\text{items.size} + 1$
(H) $x + 1$

Dynamic programming

16. Consider using dynamic programming to improve the efficiency of the following function:

```javascript
function f(a, b, c, d, e) {
    if (a == 0) return 0;
    if (a < 0) return -INFINITY;
    if (d == e) return 0;
    return max(
        f(a, b, c, d+1, e),
        f(a-b[d], b, c, d, e) + c[d]
    );
}
```

What would be the dimensionality of the dynamic programming table?

(A) 3
(B) 1
(C) 2
(D) 6
(E) 4
(F) 5

17. Consider using dynamic programming to improve the efficiency of the following function:

```javascript
function f(a, b, c, d, e) {
    if (a == 0) return 0;
    if (a < 0) return -INFINITY;
    if (d == e) return 0;
    return max(
        f(a, b, c, d+1, e),
        f(a-b[d], b, c, d, e) + c[d]
    );
}
```

How would the dynamic programming table be filled, using $a$ as an index?

(A) larger $a$ to smaller $a$
(B) smaller $a$ to larger $a$
(C) $a$ is not used as an index
18. Consider using dynamic programming to improve the efficiency of the following function:

```java
function f(a,b,c,d,e) {
  if (a == 0) return 0;
  if (a < 0) return -INFINITY;
  if (d == e) return 0;
  return max(
    f(a,b,c,d+1,e),
    f(a-b[d],b,c,d,e) + c[d]
  );
}
```

How would the dynamic programming table be filled, using \( b \) as an index?

(A) smaller \( b \) to larger \( b \) 
(B) larger \( b \) to smaller \( b \) 
(C) \( b \) is not used as an index

19. Consider using dynamic programming to improve the efficiency of the following function:

```java
function f(a,b,c,d,e) {
  if (a == 0) return 0;
  if (a < 0) return -INFINITY;
  if (d == e) return 0;
  return max(
    f(a,b,c,d+1,e),
    f(a-b[d],b,c,d,e) + c[d]
  );
}
```

How would the dynamic programming table be filled, using \( d \) as an index?

(A) smaller \( d \) to larger \( d \) 
(B) larger \( d \) to smaller \( d \) 
(C) \( d \) is not used as an index
20. Consider using dynamic programming to improve the efficiency of the following function:

```javascript
function f(a,b,c,d,e)
{
    if (a == 0) return 0;
    if (a < 0) return -INFINITY;
    if (d == e) return 0;
    return
    max(
    f(a,b,c,d+1,e),
    f(a-b[d],b,c,d,e) + c[d]
    );
}
```

What is wrong, if anything, about the following loop for filling out the dynamic programming table?

```javascript
for (a = 0; a < max_a; ++a)
    for (d = 0; d < max_d; ++d)
    {
        if (a == 0) table[a][d] = 0;
        else if (d == e) t[a][d] = 0;
        else
        {
            var x = a-b[d];
            t[a][d] =
            max(
                t[a][d+1],
                x < 0? -INFINITY : (t[x]][d] + c[d])
            );
        }
    }
```

(A) one or more of the loop indices is incorrect  (D) the a loop goes in the wrong direction
(B) the d loop goes in the wrong direction  (E) there should only be one loop (no nesting)
(C) there should be three nested loops  (F) the table is filled out correctly

21. Consider dynamically programming the following function, what should the dimensionality of the table be?

```javascript
!!change this to an actual problem
function f(a,b,c,d,e)
{
    if(a == e) return c/d;
    else if(b < d) return 0;
    return f(a,log(2^b),c%8,d + 1,e - 1)
}
```

(A) 0  (D) 4
(B) None of these answers are correct  (E) 3
(C) 1  (F) 2

For the following questions, consider dynamically programming a function f to implement the dice rolling algorithm, which calculates total number of ways to reach a sum z, rolling x dice, with each die having y sides. So if you have five dice each with six sides and desire to find the number of ways to achieve a sum of 1, then f(6,5,1) should return 0.

22. What is the dimensionality of the table?

(A) 0  (D) 1
(B) none of these answers are correct  (E) 4
(C) 3  (F) 2
23. What is/are the parameter/parameters that fill the entries of the table?

(A)  z only
(B)  x only
(C)  x and y
(D)  z and x
(E)  y only
(F)  x, y, and z
(G)  none of these answers are correct
(H)  z and y

24. What is/are the largest index/indices into the dynamic programming table?

(A)  x, y, and z
(B)  none of these answers are correct
(C)  x + 1 and z
(D)  x
(E)  x + 1 and y + 1
(F)  x + 1, y + 1, and z + 1
(G)  z and x
(H)  x - 1, y - 1