Problem 1. ***PG Problems 392, 393, and 394.***

392:	0	1	2	3	4	5	6	7	8	9	10
1	T	T	F	F	F	F	F	F	F	F	F
2	T	T	T	T	F	F	F	F	F	F	F
3	T	T	T	T	T	T	T	F	F	F	F
4	T	T	T	T	T	T	T	T	T	T	T

393:	0	1	2	3	4	5	6	7	8	9	10
1	T	F	F	F	F	T	F	F	F	F	F
2	T	F	F	F	T	T	F	F	F	T	F
3	T	F	F	T	T	T	F	T	T	T	F
4	T	F	F	T	T	T	T	T	T	T	T

394:	0	1	2	3	4	5	6	7	8	9	10
1	T	F	T	F	F	F	F	F	F	F	F
2	T	F	T	F	T	F	F	F	F	F	F
3	T	F	T	T	T	T	F	T	F	F	F
4	T	F	T	T	T	T	T	T	T	F	T

Problem 2. ***DPV Problem 6.11.***

As the hint suggested, we want to compute $L[i,j]$ = the length of the longest common subsequence of $x_{1..i} = x_1x_2\dots x_i$ and $y_{i..j} = y_1y_2\dots y_j$. $L[0,j] = L[i,0] = 0$ since in those cases one of the strings is empty. When $i > 0$ and $j > 0$ and we know $L[i-1,j-1]$, $L[i-1,j]$, and $L[i,j-1]$. The formula for $L[i,j]$ is:

$$L[i,j] = \begin{cases} \max(L[i,j-1], L[i-1,j]), & \text{if } x_i \neq y_j; \\ \max(L[i,j-1], L[i-1,j], 1+L[i-1,j-1]), & \text{if } x_i = y_j \end{cases}$$

The idea is that if $x_i \neq y_j$, then we cannot use both x_i and y_j, so the best we can do is the max $L[i,j-1]$ (the length of the l.c.s. of $x_{1..i}$ and $y_{i..j}$ that doesn't use y_j) and $L[i-1,j]$ (the length of the l.c.s. of $x_{1..i}$ and $y_{i..j}$ that doesn't use x_i). If $x_i = y_j$, then we also have the possibility of using x_i and y_j at the end of the l.c.s. of $x_{1..(i-1)}$ and $y_{1..(j-1)}$. So here is the algorithm, which clearly runs in $O(m \cdot n)$ time.

```
function lcs(x_{1..m}, y_{1..n})
  for i ← 0 to m do L[i,0] ← 0
  for j ← 0 to n do L[0,j] ← 0
  for i ← 0 to m do
    for j ← 0 to n do
      L[i,j] ← max(L[i,j−1], L[i−1,j])
      if x_i = y_i then L[i,j] ← max(L[i,j], 1+L[i−1,j−1])
```

Problem 3. ***DPV Problem 6.18.***

To make change for u with denominations 1 through i you have the choice of using a denomination i coin or not. If you use the coin, then you have to make change for $u - x_i$ using denominations 1 through $i-1$. If you don't use the coin, then you have to make change for u using denominations 1 through $i-1$. So the equation for $C[i,u]$ is

$$C[i,u] = \begin{cases} C[i-1,u] \text{ or } C[i-1,u-x_i], & \text{if } x_i \leq u; \\ C[i-1,u], & \text{if } x_i > u. \end{cases}$$

So we can compute $C[n,v]$ by

```
for i ← 0 to n do C[i,0] ← true
for u ← 1 to v do C[0,u] ← false
for i ← 1 to n do
  for u ← 1 to v do
    if x_i ≤ u then C[i,u] ← C[i−1,u] or C[i−1,u−x_i]
           else C[i,u] ← C[i−1,u]
```

Problem 4. ***DPV Problem 6.21.***

Suppose $G = (V,E)$ is an undirected graph and $U \subseteq V$.

(a) *Claim:* If U is an independent set, then $V - U$ is a vertex cover.
Proof: We argue the contrapositive. Suppose $V - U$ is **not** a vertex cover. Then for some edge (u,v), neither u nor v is in $V - U$. Hence, $u, v \in U$. Therefore, U cannot be an independent set.

(b) *Claim:* If U is a vertex cover then $V - U$ is an independent set.
Proof: We argue the contrapositive. Suppose $V - U$ is **not** an independent set. Then for somce edge (u,v), neither $u, v \in (V-U)$. Hence, neither u nor v is in U. Therefore, U cannot be a vertex cover.

(c) From (a) and (b), U is a minimal vertex cover if and only if $V - U$ is a maximal independent set. So use the $O(|V|)$ algorithm of the book to find W, a maximal independent set of T and then return $V - W$.