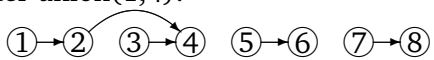
Typo fixes in **red boxes**.

Problem 1. ***DPV Problem 5.11.***

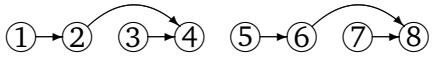
Initially:

After union(1,2), union(3,4), union(5,6), union(7,8):

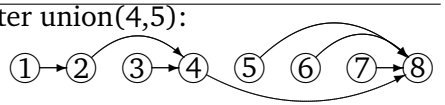
After union(1,4):



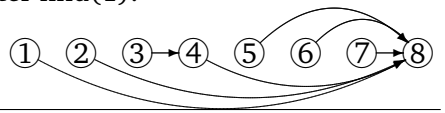
After union(6,7):



After union(4,5):



After find(1):



Problem 2. ***DPV Problem 5.13*** $A \mapsto 11$, $C \mapsto 101$, $G \mapsto 100$, $T \mapsto 0$.

Problem 3. ***DPV Problem 5.23.***

(a) Since $e \notin T$, increasing e's weight will not change the cost of a MST. So T is still a MST.

(b) Add e to T and call the graph $\widetilde{G}$. Do a DFS to find $\widetilde{G}$'s one cycle. Find the **highest** weight edge on this cycle and delete it from $\widetilde{G}$. It follows that the resulting graph is a tree and it follows from the cut property that it is a MST. This all takes $\Theta(|V|)$ time.

(c) Since $e \in T$, decreasing its weight decreases the cost of a MST and T (under $\hat{w}$) has this new weight. So T is still a MST.

(d) Delete e from T, which creates two trees T_0 and T_1. Search through E to find the lightest edge that connects T_0 to T_1. This all takes $\Theta(|V|+|E|)$ time.

Problem 4. ***DPV Problem 5.26.*** Use the union-find data structure thusly:

for $i \leftarrow 1$ **to** n **do:** $makeset(i)$
for each equality $x_i = x_j$ **do:** $union(i,j)$.
for each inequality $x_i \neq x_j$ **do:**
  if $(find(i) = find(j))$ **then return** False
return True

Problem 5. ***PG: Supplemental Problems, Page 45, Problem 3.***

Notation: $[a,c] = \{\, b \in \mathbb{R} : a \le b \le c \,\}$.

(a) The algorithm:

$\mathcal{I} \leftarrow \emptyset$; $pts \leftarrow$ the list $x_1, \ldots, x_n$
while pts is not empty **do:**
  $x \leftarrow \min(pts)$; $\mathcal{I} \leftarrow \mathcal{I} \cup \{\, [x, x+1] \,\}$.
  $pts \leftarrow$ (pts with all numbers in $[x, x+1]$ removed)
return $\mathcal{I}$

(b) CORRECTNESS PROOF: Suppose $\{\, I_1, \ldots, I_m \,\}$ are the covering intervals that the algorithm returns and $\{\, J_1, \ldots, J_k \,\}$ is an optimal cover. Assume the I_i's and J_i's are ordered by their left endpoints. We need to show $m = k$.

Claim 1: $I_1, J_2, \ldots, J_k$ is an optimal cover. *Proof:* $I_1 = [x_1, x_1+1]$ and $J_1 = [y, y+1]$ for some y. We must have $y \le x_1$, otherwise J_1 does not cover x_1. Hence, I_1 covers all the points that J_1 does and so $I_1, J_2, \ldots, J_k$ works as an optimal cover.

Claim 2: For $\ell = 2, \ldots, k$: $I_1, \ldots, I_\ell, J_{\ell+1}, \ldots, J_k$ is an optimal cover. *Proof:* This is just a repeat of the argument for Claim 1 (+ induction).

Therefore, $m = k$ as required.

Problem 6. ***PG Problems 395, 396, and 397.***

Here are some possible answers.

395: $s_1 = 3$, $s_2 = 2$, $s_3 = 2$, and $S = 4$.
396: $s_1 = 1$, $s_2 = 2$, $s_3 = 2$, and $S = 4$.
397: $s_1 = 3$, $s_2 = 2$, $s_3 = 2$, and $S = 4$.

Problem 7. ***DPV, Problem 6.1.*** *Observation:* The continuous subsequence maximum sum (c.s.m.s.) ending at position j is either just a_j or else is $a_j +$ the c.s.m.s. ending at position $j-1$.

function csms(a[1..n])
  declare $s[0..n]$ and set $s[0] \leftarrow 0$.
  for $i \leftarrow 1, \ldots, n$ **do:** $s[i] \leftarrow \max(s[i\,\mathbf{-1}] + a[i], a[i])$
  $m \leftarrow$ the maximum value in $s[0..n]$
  $k \leftarrow$ the smallest i with $s[i] = m$
  $ans \leftarrow$ the empty list
  while $\boldsymbol{m} > 0$ **do:**
    add $\boldsymbol{a[k]}$ to the front of ans; $m \leftarrow m - a[k]$; $k \leftarrow k-1$
  return ans