Feel free to use the empty space provided in this homework for filling up your answers in submitting this homework.

1. (18 points)

We are given two programs A and B that use two different implementations of the union-find data structure. Program A applies path compression when it performs each find() operation; whereas Program B does not apply path compression for its find()s.

Both programs start off with the same initial union-find data structure shown below:\n
```
                a
               / \  
              d   g  b
             /   / \  /
            k   f   h c
           / \    /  
          y   x   
```

And both programs perform the same sequence of find() operations (in the specified order):

\[ \text{find}(y), \text{find}(x), \text{find}(k), \text{find}(y), \text{find}(x), \text{find}(h) \]

Calculate the number of steps each of the above find() operations takes to climb from the element being searched to the root node. For example, the number of such steps if one were to perform a find(d) on the above shown initial tree will be 1 (under both programs). Give your answer by filling the number of steps for each find() operation in the table below:

\[ \text{Note that this union-find data structure contains only one tree in its forest and element } a \text{ is its root. Also, note that I have assigned character labels for each element which is okay as long as you work with the tree representation (and not the vector implementation) for this example.} \]
<table>
<thead>
<tr>
<th>Program A (w/ path compression)</th>
<th>Program B (w/o path compression)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. \textit{find}(y)</td>
<td></td>
</tr>
<tr>
<td>2. \textit{find}(x)</td>
<td></td>
</tr>
<tr>
<td>3. \textit{find}(k)</td>
<td></td>
</tr>
<tr>
<td>4. \textit{find}(y)</td>
<td></td>
</tr>
<tr>
<td>5. \textit{find}(x)</td>
<td></td>
</tr>
<tr>
<td>6. \textit{find}(h)</td>
<td></td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
</tr>
</tbody>
</table>

Also in the empty space provided below, show the final trees resulting from both programs, after the last find operation (i.e., \textit{6. find}(h)).

Program A’s output union-find tree:

Program B’s output union-find tree:
2. (15 points)

Starting with the union-find data structure shown below, show the sequence of union-find data structures that result from applying the following operations (in that order):

\text{union}(1, 2), \text{union}(3, 4), \text{union}(4, 5), \text{union}(6, 8), \text{union}(5, 8), \text{union}(1, 6), \text{union}(7, 9), \text{union}(10, 11), \text{union}(11, 9), \text{union}(1, 11).

\[ \begin{array}{ccccccccccc}
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 \\
\end{array} \]

Answer this question for each of the three following parts separately:

a) The \textit{unions} are performed by height (same as union-by-rank) and finds are simple;

b) The \textit{unions} are performed by size and finds are simple;

c) The \textit{unions} are performed by height and finds use path compression.

Note: There could be more than one correct answer sequence for each part. You just need to give one.