Write a recursive function **max\_path\_sum()** that takes the root of the binary tree where each node contains a positive integer value. The function should traverse the tree from the root node to all leaf nodes and return the maximum path sum. *Note: A path sum is defined as the total sum of node values from the root to a leaf. A leaf node is a node that has no left or right child.*

[You are not allowed to use any global variables or built-in functions. You can use multiple parameters and helper functions]

| Input Tree | Output | Explanation |
| --- | --- | --- |
|  | 25 | Sum of node values for each root-to-leaf path and find the maximum path sum.  Paths and their Sums:   * 1 → 7 → 2 = 1 + 7 + 2 = 10 * 1 → 7 → 6 → 5 = 1 + 7 + 6 + 5 = 19 * 1 → 7 → 6 → 11 = 1 + 7 + 6 + 11 = 25 * 1 → 9 → 9 → 5 = 1 + 9 + 9 + 5 = 24   The maximum path sum is 25, which comes from the path  1 → 7 → 6 → 11. |

Write a recursive function **min\_path\_sum()** that takes the root of the binary tree where each node contains a positive integer value. The program should traverse the tree from the root node to all leaf nodes and return the minimum path sum. *Note: A path sum is defined as the total sum of node values from the root to a leaf. A leaf node is a node that has no left or right child.*

[You are not allowed to use any global variables or built-in functions. You can use multiple parameters and helper functions]

| Input Tree | Output | Explanation |
| --- | --- | --- |
|  | 10 | Sum of node values for each root-to-leaf path and find the maximum path sum.  Paths and Their Sums:   * 1 → 7 → 2 = 1 + 7 + 2 = 10 * 1 → 7 → 6 → 5 = 1 + 7 + 6 + 5 = 19 * 1 → 7 → 6 → 11 = 1 + 7 + 6 + 11 = 25 * 1 → 9 → 9 → 5 = 1 + 9 + 9 + 5 = 24   The minimum path sum is 10, which comes from the path 1 → 7 → 2. |