Solution 1

```java
// Copyright © 2020 AlgoExpert, LLC. All rights reserved.

import java.util.*;
import java.util.stream.Collectors;

class Program {
  // O(n) time | O(n) space - where n is the length of the path
  public static String shortenPath(String path) {
    boolean startsWithPath = path.charAt(0) == '/';
    String[] tokensArr = path.split("/");
    List<String> tokensList =  Arrays.asList(tokensArr);
    List<String> filteredTokens = tokensList.stream().filter(token -> isImportantToken(token)).collect(Collectors.toList());
    List<String> stack = new ArrayList<String>();
    if (startsWithPath) stack.add("");
    for (String token : filteredTokens) {
      if (token.equals("..")) {
        if (stack.size() == 0 || stack.get(stack.size() - 1).equals("..")) {
          stack.add(token);
        } else if (!stack.get(stack.size() - 1).equals("")) {
          stack.remove(stack.size() - 1);
        } else {
          stack.add(token);
        }
      }

    if (stack.size() == 1 && stack.get(0).equals("")) return "/";
    return String.join("/", stack);
  }

  public static boolean isImportantToken(String token) {
    return token.length() > 0 && !token.equals(".");
  }
}

```