FP树的实现思路：

1. 读入所有数据，遍历所有事务，找出所有项的支持度，存入哈希表。
2. 再次遍历所有事务
   1. 对每条事务里的项按照哈希表中的项全局支持度排序。
   2. 遍历当前事务的项，从根结点构建FP树。
3. 使用构建出的FP树进行数据挖掘
   1. 检查是否为单路径树
      1. 如果是，直接对所有节点进行组合，共2^n种可能，每种组合与“父项集”的并集作为频繁项集输出，支持度为组合中节点的最小支持度。该层数据挖掘结束。
   2. 按照全局支持度排序所有项
   3. 遍历排序后的所有项
      1. 从该项第一次出现的节点向根节点回溯，构建支持度为该节点支持度的频繁模式基（不含自身）。
      2. 找出该节点的下一个同值节点，重复上述步骤。
      3. 将该项加入“父项集”，将该“父项集”本身作为频繁项集输出，支持度为该项的全局支持度。然后使用上述带支持度的频繁模式基集合建立新的条件FP树，并递归进行数据挖掘。

感想：

在编写此算法的过程中，我使用了C#语言进行编程，由于C#作为高级语言自己实现了许多优化，并提供了高效的哈希表等数据结构和快速排序算法，再加上面向对象编程直观简洁，编程过程十分顺畅，并且额外实现了很多功能。