三、实验内容及基本要求

（四）文本索引

编写一个构建大块文本索引的程序，然后进行快速搜索，来查找某个字符串在该文本中的出现位置。

你的程序应该使用两个文件名作为命令行参数：文本文件（我们称为语料库）和包含查询的文件。 假设这两个文件只包含小写字母、空格和换行符，查询文件中的查询由换行符分隔。 这不是一个限制，因为你可以使用一个过滤器将任何文件转换为此格式。

你的程序应该读取语料库，将其存储为（可能巨大）字符串，并可能为其创建索引，如下所述。 然后它应该逐个读取查询（假设在命令行中的第二个命名文件中，每行有一个查询），并打印出语料库中每个查询在文本文件中首次出现的位置。对于由如下内容构成的 corpus 文件

it was the best of times it was the worst of times it was the age of wisdom it was the age of foolishness it was the epoch of belief it was the epoch of incredulity it was the season of light it was the season of darkness it was the spring of hope it was the winter of despair

以及如下内容构成的 query 文件

wisdom
season
age of foolishness
age of fools

查询结果如下：

18 wisdom
40 season
22 age of foolishness
-- age of fools

解决这个问题有很多种不同的方法，这些方法在实现方便性，空间要求和时间要求方面都有不同的特点。 此任务的一部分是吸收此信息，以帮助你确定使用哪种方法以及如何将其应用于此特定任务。

你可以从本书中基于程序 3.15 的蛮力搜索实现开始。 也就是说，不建立索引：只需搜索每个查询字符串的语料库即可。 如果语料库很小或者查询不多，这个解决方案是很好的。但是，当语料库庞大且查询量大的时候，这种方法太慢了，因此，你需要实现一个更快的解决方案。

一种快速搜索的方法是在语料库（每个字符位置一个指针）上进行指针排序，然后使用折半搜索。 如果你想采用这种方法，可以使用标准 C 库中的 qsort 和 bsearch 函数。 这种方法的主要挑战是完全理解程序 3.17; 开发一个类似的程序来构建指针索引，按照排序顺序访问关键字（如图 3.13 所示）；并找出调用 bsearch 的必要接口使用索引执行查询。 特别地，对于进行排序和搜索你需要适当的“比较”功能（不同的!）。

另一种可能的方法是从程序 12.10 开始。 这段代码基本上提供了一个完整的解决方案，但为了使其正常工作，你必须进行一些小的更改，因为它们在许多细节上与此处指定的问题不同，并且因为缺少各种小的代码。 你可能需要编辑此代码，或从头开始编写自己的代码。 再次，必须仔细考虑“比较”功能。

你可以从这个网站 pizzachili.dcc.uchile.cl/texts.html 或 http://corpus.canterbury.ac.nz/descriptions/上下载 corpus 数据测试你的程序。

鼓励修改你的程序使其能够计算出每次查询串在 corpus 中出现的次数。

注：本题章节是指该书的 C 语言版