计算机实验室4

CSCI 1913:介绍算法、数据结构和程序开发

1介绍

假设一个单词中python没有字典数据结构。字典数据结构是许多问题的核心，因此这自然是一个主要问题。大多数程序员会通过编写函数来模拟字典的行为来应对这种情况。这就是我们将要采取的方法——我们将“假装”dictionary不是用Python实现的，然后编写函数来实现我们自己的。

在这个实验室中，我们将使用两个不同版本的“关联列表”结构来模拟字典。在关联列表中，每个键-值对存储在一个元组中，键-值对的集合存储在一个列表中。(即关联存储在一个列表中，因此称为关联列表)。

我们将探讨两种方法:

1.未排序的关联列表

2.已排序的关联列表

在未排序的关联列表中，关联可以按任何顺序排列，并且可以使用线性搜索在关联列表中搜索键。这导致了非常简单的代码，它在O(n)时间内运行。

在已排序的关联列表中，关联按键排序存储。这需要更仔细的代码，但可能更有效。特别是，通过使用二分查找来查找键值对，我们可以将大多数操作减少到O(log(n))的运行时间。

本实验室将:

•给你更多使用列表和元组的练习

让你更好地理解字典

•让你练习实现线性和二进制搜索。

1

2软件环境设置

与前面的实验一样，您将进行python编程。虽然您可以在任何合理的编程环境中准备python程序，但我们仍然推荐PyCharm。基本的PyCharm设置说明可以在之前的实验中找到。

1.您将需要为这个新实验室创建一个新的PyCharm项目。在一个项目中混合多个实验室可能会不小心提交错误的文件或其他可能影响评分的“结转”。

2.如果PyCharm要求您设置一个项目解释器，但是没有默认的显示，那么您应该能够添加一个。在CSELabs机器上，“/usr/bin/python3”是默认的python3解释器的名称，这是一个不错的选择。

3.记住，要注意PyCharm是在何处创建python项目本身的，您甚至可能希望确保在设置项目时将其放置在为此类指定的位置。最终，您需要在PyCharm之外找到这些文件。

3实验室提醒

•如果你愿意，你可以每周更换实验室合作伙伴，你从来没有被限制在一个合作伙伴中，如果你需要帮助寻找合作伙伴，请向助教咨询

•伙伴关系中的每个程序员应该是平等的——尽量不要跑在你的伙伴前面，确保他们理解你在做什么，并确保每个程序员花在键盘上的时间是平等的。(记住，真正的结对编程是两个程序员，一个键盘，一个屏幕)。

•如果你没有完成，你和你的合作伙伴应该继续在实验室外合作完成实验。我认为两个学生在学术上都对你的解决方案负责(我希望任何一个学生都能向我解释他们的代码)。

4个文件

这个实验室将涉及以下文件

•assoc\_lists。你应该写这个模块。这是测试所期望的名称。

•assoc\_list\_test。提供了这个文件。它对所有需要的功能都进行了测试。它可以在画布上找到，将作为我们的评分过程的基础，以及一个有用的工具，为您的个人发展。

2

软件设计

5

在python中，我们将使用具有两个元素的元组来表示键和值之间的关联，第一个元素是值，第二个元素是键。虽然这个选择是任意的(即基本上任何表示和顺序都可以)，但这可能不是表示的首选，所以在编码时要小心，以免混淆这些元素的顺序。

我们将存储两种类型的关联列表——线性列表(未排序)和二进制列表(按键值排序)。

所以下面这个字典:

bin\_list ={“乌鸦”:“乌鸦”:3日,49岁的“鱼”:2,“猫”:13日

}

表示为线性关联表:

[(3，“乌鸦”)，(49，“乌鸦”)，(2，“鱼”)，(13，“猫”)]，将被表示为这样一个二进制列表:

“猫”[(13日),(3,“乌鸦”),(2,'鱼'),(49“乌鸦”)]

我们将没有限制的值对象,但是,由于需要,我们将只使用关键对象可比用python数据类型(如数据类型允许您使用<和>),这意味着我们可以用数字(整数或浮点)或字符串。

我们需要的函数是基于dictionary类的核心操作:

•contains(dict, key)返回True或False，表示该键是否在字典中的关联中。

•get(dict, key)，返回与字典中的键相关联的值(如果可能的话)，或者python中的特殊值None(如果key不在字典中)(注意，None没有引号，它是一个特殊值，不是字符串)

•put(dict, key, value)修改字典，使给定的键现在与给定的值相关联。如果键已经与某个值关联，则应将旧值更新为新值。如果键与字典中的值没有关联，则应该将新的值关联添加到字典中。

测试文件中的示例可能有助于您更好地理解这些函数

需要的。

3.

所需的功能

6

这个实验需要六个函数，三个表示线性搜索(未排序)关联列表，三个表示二进制搜索(排序)关联列表。

lin\_contains(lin\_dict, key)包含用于线性(未排序)关联列表的函数。预计在O(n)时间内运行

•lin\_get(lin\_dict, key)获取用于线性(未排序)关联列表的函数。预计在O(n)时间内运行

•lin\_put(lin\_dict, key, value)用于线性(未排序)关联列表的put函数。预计在O(n)时间内运行

•bin\_contains(bin\_dict, key)包含用于二进制(排序)关联列表的函数。期望在O(log(n))时间内运行

•bin\_get(bin\_dict, key)获取用于二进制(排序)关联列表的函数。期望在O(log(n))时间内运行

•bin\_put(bin\_dict, key, value)函数，用于二进制(排序)关联列表。预计在O(n)时间内运行。注意，这个函数是用来维护bin\_dict的排序顺序的。

三个主要函数(包含、获取、放置)的行为如上所述。get和contains方法需要返回值行为，但是put函数不需要。只要对关联列表进行了预期的更改，就可以从put函数中返回您想要的任何内容。

上面列出的big-O运行时行为是这段代码的要求，三个lin函数应该在线性O(n)时间内运行。bin get和bin包含的函数需要使用二进制搜索算法并在O(log(n))时间内运行。(仅仅使用二分查找是不够的，如果您出于任何原因遍历整个后记，您仍然会有O(n)运行时。)

bin put必须保持bin dict的排序顺序，这意味着如果它需要添加一个新键，它将需要找出键在列表中的位置。虽然用二分查找在技术上是可行的，但是它需要一个我们没有讨论的修改。因此，我们不需要在bin put函数中使用二分查找。不过，这不是主要的问题，因为向列表中间添加元素是一个O(n)过程——因此，在O(log(n))中不可能实现这个函数。

如果需要，您可以参考课本的二进制搜索代码，但是您应该提前知道，如果不进行修改，这些代码将无法工作，因为关联被存储为元组的有序值，然后是键。但是，您应该尝试根据内存/第一原则来编写这个算法。虽然这有点难，但如果你这样做了，你会从实验室学到更多。

4

7日交付

在提交您的工作之前，请使用测试来仔细检查您的代码是否没有bug，是否具有预期命名的每个函数，以及是否返回与预期兼容的数据类型。

将提交文件命名为assoc\_lists。确保最上面的评论里有你的名字，如果你和搭档一起工作，那就写上搭档的名字。如果你没有合作伙伴，而他们的名字不在你的档案上，他们将得不到学分，或者，你将被指控作弊(取决于他们是否提交了档案)

您应该只提交assoc\_lists。py文件。您将通过canvas提交此文件。如果你和实验室合作伙伴一起工作，两个学生都应该提交一份副本。这与前几周不同，我们相信如果每个学生都上交自己的论文，将更容易快速地对提交的论文进行评分。

每个学生都要交一份实验报告!即使他们与合作伙伴一起工作。如果您和您的合作伙伴提交相同的文件是可以的。我们只是想让每个人都提交一份文件。

这将在下周你们的下一个实验开始之前完成。具体时间将根据您的实验室的正式开始时间。Canvas将接受重新提交，也将接受延迟提交。请务必不要因为意外而迟交作业。

如果你在实验时间内完成了这个实验，请随时通知你的实验助教，然后早点离开。如果你在实验期间没有完成这个实验，你有责任在下一个实验之前完成。

5