

**Lab Report**

**实验报告**

|  |  |
| --- | --- |
| **Course**: | Class Libraries and Data Structures |
| **Semester**: | 1st semester of the academic year **2023-2024** |
| **Major**: | Software Engineering |
| **Class**: | 2022 |
| **Student Name**: |  |
| **Student ID:** |  |
| **Teacher:** | ZHAO, Hengjun (赵恒军) |

**School of Computer and Information Science**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name | | Time Complexity and Runtime Analysis  时间复杂度和C++运行时分析 | | | |
| Date | | Oct，2023 | Type | | ☑Confirmatory （验证确认型）  ☑Design（设计型）  🗆Comprehensive（综合型） |
| 1. **Objective & Requirements（实验目的）**    1. Understand the theoretical time complexity of an algorithm and know how to analyze it   理解算法理论复杂度的概念和表示方法，掌握算法理论复杂度的分析方法   * 1. Grasp the use of random numbers and techniques for measuring execution time in C++   掌握在C++中使用随机数生成器和计时工具进行算法运行时时间复杂度的测量方法   * 1. Grasp runtime analysis of programs to show the effect of theoretical complexity on the time cost of real programs by running programs and measuring the time cost   掌握对大量实验数据进行统计和理论分析的技术，能够从实验上验证算法的理论时间复杂度 | | | | | |
| 1. **Experimental environment (**platform and software**)（实验环境）**   Windows 7 (or higher versions) + Visual Studio 2010 (or higher versions) | | | | | |
| 1. Experimental content and design (Main Content, Procedure, Codes and Results)（此部分应包含每一个实验内容的详细设计，含实验思路、详细实验步骤、核心代码说明等） 2. Task 1 （任务1）   You are provided with a template container based on singly linked list. Please read the source code and implement a new method addTail() that can add a new element at the end of the linked list. Note that your are NOT allowed to modify any existing codes of the lined container, including adding new data or function member.  请阅读并理解所给单向链式模板容器，为其实现一个新的元素添加方法addTail()以实现在链式容器的末尾添加新的元素。注意，在实现addTail()方法时，请不要修改该容器中任何已经存在的代码，包括添加新的数据或方法成员。  **实验思路：**  从末尾添加新元素，也就是让最后一个节点的next指针指向新节点，那么只需要在方法内设立一个oldTail，让它每次都从头迭代到尾，这样就获得了一个指向尾结点的指针。再考虑一下当容器为空的时候，让新节点作为第一个结点，让头指针指向新节点就好了。代码以及详细说明如下：     1. Task 2 （任务2）   Please rewrite the input() method for the Employee class. The new input() method takes a parameter int id, and set the calling employee with name “employee+id”, e.g. “employee123”, and with a randomly generated integer as the grosspay.  为Employee类改写input方法。新的input方法接收一个整型参数id，将调用者的姓名置为“employee+id”，例如“employee123”，同时将其总收入置为一个随机生成的整数。  **实验思路：**  直接用string类封装好的方法就好了，再使用inputEmployee….中初始化的srand，调用rand函数生成随机数来给总收入赋值。     1. Task 3 （任务3）   Please implement two methods for the Company class i.e.   * + - * void inputEmployeeHead(int total\_num);       * void inputEmployeeTail(int total\_num);   The integer total\_num is the method argument that specifies the total number of employee to input. The information of each input employee is set by the method in Task 2. inputEmployeeHead() is based on addHead() and inputEmployeeTail() is based on addTail() that is implemented in Task 1.  为公司类实现inputEmployeeHead()和inputEmployeeTail()方法，这两个方法均接收一个整型数total\_num为参数，表示待录入信息的员工的总数；每一个员工的信息使用任务2的input方法录入；要求inputEmployeeHead()基于容器类的addHead()方法实现，inputEmployeeTail()基于上述任务1（Task 1）实现的容器类的addTail()方法实现。  实验思路：  循环输入员工信息，每一次将信息放入容器，其id随便，我就让它按顺序排列吧。两个函数唯一的区别就是一个调用addHead从头添加，一个调用addTail从尾添加。还要提前为input函数中要随机化的grosspay初始化一个随机数生成器，就取现在的时间作为seed。     1. Task 4 （任务4）   Using runtime analysis to measure the time costs of inputEmployeeHead and inputEmployeeTail by increasing the total number of employee total\_num, e.g., 1000, 2000, …, 10000, 20000, …, 100000, and so on. Record the time costs of the two methods for each value of total\_num. Plot the data in a figure and try to fit the data using a curve (数据拟合，曲线拟合).  Analyze the theoretical time complexity of inputEmployeeHead and inputEmployeeTail. Compare your theoretical analysis to the experimental data you obtained.  使用运行时分析工具统计inputEmployeeHead 和 inputEmployeeTail方法的耗时；设置不同的参数测试并记录两个算法的多组时间数据。使用所得数据绘图并进行数据拟合，比较你获取的实验分析结果和两个算法的理论复杂度分析结果。  **答：**  Main函数实现如下：  先记录开始时间，等待全部员工信息存入完毕记录结束时间，输出时间差。这里我创立一个循环，每一次作为一次测试，每一次测试都记录存入的员工总数和所用时间，梯度为5000，总共十次测试。采用动态开辟内存的方法，每一次测试完毕就释放掉当前容器。代码如下：    先后统计使用inputEmployeeHead 和 inputEmployeeTail方法的耗时，得到两组数据  inputEmployeeHead方法：    [8,16,26,32,40,47,54,62,71,80]  inputEmployeeTail方法：    [62,241,491,928,1592,2343,3266,4580,5724,7114]  理论时间复杂度：  Head：存入多少个员工，就执行多少次调整指针的操作，所以时间复杂度为O(n)  Tail：由于每次从尾部加入新节点都要使一个指针从头迭代到尾，存入员工数量越多，指针迭代次数越多，插入第n个结点时要迭代n次，相当于1+2+3+…+n，所以时间复杂度为O(n²)  实验结果分析：  使用python编写，使用numpy中的ployfit函数拟合曲线并用ployld函数生成函数。  数据用拟合图像及函数：  inputEmployeeHead：  拟合函数：1.559 x + 0.7333    inputEmployeeTail：  拟合函数3.272 x² - 22.29 x + 98.22    显然，在实验中，inputEmployeeHead方法的时间复杂度为O(n)，inputEmployeeTail方法的时间复杂度为O(n²) | | | | | |
| 1. **Result analysis and discussion**（Analysis of experimental results and summing up the harvest and the existing problems）（此部分应包含实验结果，对实验结果的分析，实验收获的总结，实验中存在问题的讨论等；另外，需要回应一下如下思考题：1. Lab1-2要求实现一个双向链表和相应的迭代器。本实验不允许使用双向链表，只能使用单向链表，为什么？2. 在本实验中，员工的总收入是随机生成的，为此需要设置随机数生成器的种子（seed），请思考这个设置放置在代码的什么位置为好？为什么？）   实验中，inputEmployeeHead方法的时间复杂度为O(n)，inputEmployeeTail方法的时间复杂度为O(n²)，与理论计算值一致。  通过这次实验，我基本掌握了时间复杂度的计算方法，顺便复习了链表相关知识，最重要的是学会了用计算时间差的方式计算程序执行的时间，并练习了用其他工具（python）来拟合曲线函数。  思考题：   1. 如果使用双向链表，那么从头加入结点与从尾加入的操作差别不大，时间复杂度相同，均为O(n)，那么这个实验结果就基本一样了呀，还是不用双向链表比较有挑战性。、 2. 我觉得可以放在inputEmployeeHead和inputEmployeeTail方法中，这样可以避免每一次传入员工调用input时都随机化这个生成器（浪费时间），而每一次既保证了代码的效率，又不失其随机性。 | | | | | |
| Comments & Evaluation | Content & Design (A-E) | | |  | |
| Procedure & Codes (A-E) | | |  | |
| Results (A-E) | | |  | |
| Analysis & Discussion (A-E) | | |  | |
| Score (A-E):  Feedback comments: | | | | |