

**Lab report**

|  |  |
| --- | --- |
| **Course**: | Class Libraries and Data Structures |
| **Semester**: | 1st semester of the academic year **2020-2021** |
| **Major**: | Software Engineering |
| **Class**: | 2019 |
| **Student Name**: | 冯春霖 |
| **Student ID:** | 222019321062074 |
| **Teacher:** | ZHAO, Hengjun (赵恒军) |

**School of Computer and Information Science**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name | | Iterator and Time Complexity | | | |
| Date | | Nov 27，2020 | Type | | √ Confirmatory  √ Design  □Comprehensive |
| 1. **Objective & Requirements**    1. Know the use of iterator and understand its implementation details    2. Can use iterator to traverse a list in the reverse order to finish a certain task    3. Understand the time complexity and know how to analyze it; show the effect of theoretical complexity on the time cost of real programs by running programs and measuring the time cost | | | | | |
| 1. **Experimental environment (**platform and software**)**   Windows 7 (or higher versions) + Visual Studio 2010 (or higher versions) | | | | | |
| 1. **Experimental content and design** (Main Content, Procedure, Codes and Results)   Task 1   * Using the source code sent to you about the linked list template with iterator, implement three methods   + operator--(int) for the iterator inner class   + Begin() for the container class   + End() for the container class   + Based on the container and iterator class, implement the method findBestPaidReverse() for the Company class that find the best paid employee by traversing the list of employees in the reverse order * Test your implementation in the main() method   Task 2   * Based on the source code sent to you, implement   the following two methods for the Company class   * + findBestPaid()   + findBestPaidReverse()   **!!!**Note that the container class is implemented in Task 1 so is not provided here**!!!**   * Test your implementation in main() and measure the time   costs of both methods by setting the two constants in  company.h   * + NUM\_EMPLOYEE   + MAX\_SALARY   Task 1：  Implement of operator—():  template<class T>  typename ListTemp<T>::Iterator ListTemp<T>::Iterator::operator--(int)  {  Iterator temp = Iterator(head, head);  Iterator \_this = \*this;  while (true)  {  if (temp == \_this)  break;  \*this = temp;  temp++;  }  return \_this;  }  Implement of Begin() and End():  template<class T>  typename ListTemp<T>::Iterator ListTemp<T>::Begin() const  {  return Iterator(head, head);  }  template<class T>  typename ListTemp<T>::Iterator ListTemp<T>::End() const  {  return Iterator(head, NULL);  }  Implement of findBestPaidReverse():  void Company::findBestPaidReverse()  {  bestPaid = Employee();  ListTemp<Employee>::Iterator itr = container.End();  while (!(itr == container.Begin()))  {  itr--;  if (\*itr > bestPaid)  bestPaid = \*itr;  }  }  Test in main.cpp:    Task2：  Implement of findBestPaid() and findBestPaidReverse():  void Company::findBestPaid()  {  bestPaid = 0;  ListTemp<int>::Iterator itr = container.Begin();  while (!(itr == container.End()))  {  if (\*itr > bestPaid)  bestPaid = \*itr;  itr++;  }  }  void Company::findBestPaidReverse()  {  bestPaid = 0;  ListTemp<int>::Iterator itr = container.End();  while (!(itr == container.Begin()))  {  itr--;  if (\*itr > bestPaid)  bestPaid = \*itr;  }  }  Test in main.cpp:    Change MAX\_EMPLOYEE 50000 to 10000: | | | | | |
| 1. **Result analysis and discussion**（Analysis of experimental results and summing up the harvest and the existing problems）   This experiment deepened my understanding of iterators, and I learned about how to measure program time-used. At the same time, I gained a better understanding of time complexity, and by comparing the difference in time spent on forward and backward bestPaid lookups, I realized that designing a proper algorithm can save a lot of time. | | | | | |
| Comments & Evaluation | Content & Design (A-E) | | |  | |
| Procedure & Codes (A-E) | | |  | |
| Results (A-E) | | |  | |
| Analysis & Discussion (A-E) | | |  | |
| Score (A-E):  Feedback comments: | | | | |