

**Lab report**

|  |  |
| --- | --- |
| **Course**: | Class Libraries and Data Structures |
| **Semester**: | 1st semester of the academic year **2019-2020** |
| **Major**: | Software Engineering |
| **Class**: | 2018 |
| **Student Name**: | SONG,Xingjian（宋行健） |
| **Student ID:** | 222019321062006 |
| **Teacher:** | ZHAO, Hengjun (赵恒军) |

**School of Computer and Information Science**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name | | C++ Tempaltes | | | |
| Date | | Oct 17，2019 | Type | | √ Confirmatory  √ Design  □Comprehensive |
| 1. **Objective & Requirements**    1. Understand the concept of containers; can use template to define generic containers    2. Understand the concept of generic algorithms; use generic method to implement container template    3. Understand the difference between contiguous memory allocation and linked memory allocation; use template to implement a container with linked storage | | | | | |
| 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) 2. Task 1    1. In the template container class sent to you, implement a method remove(i) that can remove the item of index **i** in the container (Note that the first item is of index 0)    2. Based on remove(i), implement a method removeEmployee() in the company class that allows the user to input an index i and then remove the employee of index **i.** (Note that the first employee is of index 0)    3. Test your implementation in the main() function 3. Task 2    1. Implement a method for adding a new element at the head of the linked list for the container template with linked storage  * void AddHead()   1. Based on AddHead(), implement the inputEmployee() method for the company class   2. Test your implementation in the main function   Task1:  首先在company类的头文件中添加void removeEmployee()函数的声明，同时在conTemp类的头文件中添加void Remove(int i)函数的声明。removeEmployee()函数提供面向对象的提示语句，并调用remove(int i)函数，remove(int i)函数通过一次循环将第i位后面的数据前移一位。  与之前不同的地方是remove函数是有返回值的，返回一个模板类型的参数。      Task2:  首先在listTemp.h文件中添加模板函数AddHead()，创建一个新节点newHead，将需要输入的数据存入它的data中，然后将它的next指针指向原先的头指针（即原链表第一位数据的地址），最后将头指针指向新建节点的数据地址，链表长度自增1    然后在listTemp.h文件中添加有返回值的模板函数Adccess(const int n)，新建一个节点指针指向head，新建一个模板类型作为输出结果。通过while循环将节点传递到所指定的链节，返回链节中所保存的data    然后在company.cpp中类似之前写入inputEmployee()即可 | | | | | |
| 1. **Result analysis and discussion**（Analysis of experimental results and summing up the harvest and the existing problems）   Task1:  模板类和普通类的使用方法类似，需要注意的是在函数前面要加上template<class T>以表示这是一个模板函数。    Task2:  链表与数组不同的是，链表可以方便的在任意位置添加节点，但是想要访问某一节点却要从头开始遍历。  在写AddHead()函数时，要注意节点指针传递的顺序，刚开始我先把head指针指向了新建节点，报错。原因在于head被赋新值后，后面节点的地址就丢失了。应该先将后面节点的地址存在新建节点的next中，再将head指向新建节点。  在写遍历函数Access()时要注意到返回值的类型是T模板类型，刚开始我返回了一个Node的节点类型产生了错误。另外，在遍历循环的时候要留意临界情况。 | | | | | |
| Comments & Evaluation | Content & Design (A-E) | | |  | |
| Procedure & Codes (A-E) | | |  | |
| Results (A-E) | | |  | |
| Analysis & Discussion (A-E) | | |  | |
| Score (A-E):  Feedback comments: | | | | |