重庆大学本科学生实验项目任务书

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 实验题目 | | 栈、队列及其应用 | | | | | |
| 学院 | 计算机学院 | | 专业 | | 信息安全 | 年级 | 2018 |
| 实验目的   * 1. 理解栈和队列的逻辑结构以及输入输出等基本运算。 * 2. 理解用队列实现栈的原理及实现方法。 * 3. 熟练掌握使用栈识别有效的出栈顺序的方法。 | | | | | | | |
| 实验内容：   * 1. 完成下面的栈类QStack，使用其中的双队列实现入栈、出栈等基本运算   template <typename E>  class QStack : public Stack<E>{  private:  int maxSize; //栈的容量  AQueue QA；  AQueue QB; //基于数组实现的队列  public:  QStack(int size = defaultSize): QA(size), QB(size) //初始化队列  {  maxSize = size;  }  ~QStack() { }  //完成下列函数的代码  **void clear(){ }**  **void push(const E& it) { }**  **E pop() { }**  **const E& topValue() const { }**  **virtual int length() const { }**    };   * 2. 设1,2,…,N依次入栈QStack<int>, 判断由这N个整数构成的整数序列<a1,a2,…,aN> 是否为有效的出栈顺序。同时我们限定栈中只能存储K个整数(0<K<=N)，即整数入栈必须满足stack.length() < K, 如果stack.length()==K, 只能从中弹出1个以上的整数后，下一个整数才能入栈。（注：栈的容量必须大于或等于K） * 输入格式: 第一行有三个正整数N K m：N表示入栈的最大整数，K为栈中存储的整数数量上限，m表示接下来有m行输入，每一行都有1到N的整数的一组序列（空格分开）。 * 输出格式，输出m行字符T或F, 第i行的字符T(F), 表示第i行序列为有效（无效）的出栈顺序 (1<=i<=m) * 实例：   输入  5 3 3  1 2 3 4 5  3 2 1 5 4  1 5 4 3 2  输出  T  T  F     * 3. 最后提交完整的实验报告和源程序。 | | | | | | | |
| 参考资料：   * 1. Data Structures and Algorithm Analysis (C++ Version) Clifford A. Shaffer   + 2. Data Structure and Algorithm Analysis in C++ (Third Edition)，Mark Allen Weiss， Pearson Education, 2006.   + 3. Data Structures, Algorithms, and Applications in C++，Sartaj Sahni， McGraw-Hill, 1998.   + 4.《数据结构（ C 语言版）》，严蔚敏，吴伟民编著，清华大学出版社，2007年第1版 | | | | | | | |
| 任务下达日期 2019 年 10月 28日 | | | | 完成日期 年 月 日 | | | |

说明：学院、专业、年级均填全称，如：计算机学院、计算机科学与技术、2018。

**《数据结构》实验报告**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **年级、专业、班级** | | **2018级计算机学院信息安全专业1班** | | | **姓名** | **周泽峻** |
| **实验题目** | 栈、队列及其应用 | | | | | |
| **实验时间** | **2019/11/4** | | **实验地点** | **DS1408** | | |
| **实验成绩** |  | | **实验性质** | **□验证性** √**设计性 □综合性** | | |
| 教师评价：  □算法/实验过程正确； □源程序/实验内容提交 □程序结构/实验步骤合理；  □实验结果正确； □语法、语义正确； □报告规范；  其他：  评价教师签名： | | | | | | |
| 一、实验目的   * 1. 理解栈和队列的逻辑结构以及输入输出等基本运算。 * 2. 理解用队列实现栈的原理及实现方法。 * 3. 熟练掌握使用栈识别有效的出栈顺序的方法。 | | | | | | |
| 二、实验项目内容   * 1. 完成下面的栈类QStack，使用其中的双队列实现入栈、出栈等基本运算   template <typename E>  class QStack : public Stack<E>{  private:  int maxSize; //栈的容量  AQueue QA；  AQueue QB; //基于数组实现的队列  public:  QStack(int size = defaultSize): QA(size), QB(size) //初始化队列  {  maxSize = size;  }  ~QStack() { }  //完成下列函数的代码  **void clear(){ }**  **void push(const E& it) { }**  **E pop() { }**  **const E& topValue() const { }**  **virtual int length() const { }**    };   * 2. 设1,2,…,N依次入栈QStack<int>, 判断由这N个整数构成的整数序列<a1,a2,…,aN> 是否为有效的出栈顺序。同时我们限定栈中只能存储K个整数(0<K<=N)，即整数入栈必须满足stack.length() < K, 如果stack.length()==K, 只能从中弹出1个以上的整数后，下一个整数才能入栈。（注：栈的容量必须大于或等于K） * 输入格式: 第一行有三个正整数N K m：N表示入栈的最大整数，K为栈中存储的整数数量上限，m表示接下来有m行输入，每一行都有1到N的整数的一组序列（空格分开）。 * 输出格式，输出m行字符T或F, 第i行的字符T(F), 表示第i行序列为有效（无效）的出栈顺序 (1<=i<=m) * 实例：   输入  5 3 3  1 2 3 4 5  3 2 1 5 4  1 5 4 3 2  输出  T  T  F     * 3. 最后提交完整的实验报告和源程序。 | | | | | | |
| 三、实验过程或算法（源程序）   * 1.Queue的实现：   （其中，topValue函数去掉了引用和常成员函数限制，因为1)Queue中的元素会被弹出，返回引用会有风险。2)对QA和QB进行了enqueue和dequeue操作，不满足常成员函数的条件。）           * 2.判断是否为有效的出栈顺序 | | | | | | |
| 四、实验结果及分析和（或）源程序调试过程  输出结果： | | | | | | |