计算机科学技术学院实验报告

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **课程名称** | 面向对象程序设计 | | | **学 号** | 2405111B08 |
| **实验项目** | 实验一 - 用数组实现线性表 | | | **姓 名** | 王子睿 |
| **学 时** | 2 | **项目性质** | 综合型 | **班 级** | 2405111B |
| **指导教师** | 蒋振刚 | **实验地点** | 南实训424 | **日 期** | 2025年3月24日 |
| 1. **实验目的和要求**   实验目的：练习类的设计和使用。体会抽象数据类型的内涵。  实验要求：设计和实现一个线性表类SList，满足：  1）用数组实现存储；  2）实现线性表的基本操作；  3）测试代码验证 List的基本操作。 | | | | | |
| 1. **实验环境**   Visual Studio Code  MinGW-w64 | | | | | |
| 1. **实验内容与过程**   #include <iostream>  class SList  {  private:      int \*data;      int maxSize;      int currentSize;  public:      SList(int size)      {          maxSize = size;          currentSize = 0;          data = new int[maxSize];      }      SList(const SList &list)      {          maxSize = list.maxSize;          currentSize = list.currentSize;          data = new int[maxSize];          for (int i = 0; i < currentSize; i++)          {              data[i] = list.data[i];          }      }      ~SList()      {          delete[] data;      }      void insert(int index, int value)      {          // Range check          if (index < 0 || index > currentSize)          {              std::cout << "Index out of range." << std::endl;              return;          }          // Check if the list is full          if (currentSize == maxSize)          {              std::cout << "List is full." << std::endl;              return;          }          // Move elements          for (int i = currentSize; i > index; i--)          {              data[i] = data[i - 1];          }          // Insert value          data[index] = value;          currentSize++;      }      void remove(int index)      {          // Range check          if (index < 0 || index >= currentSize)          {              std::cout << "Index out of range." << std::endl;              return;          }          // Move elements          for (int i = index; i < currentSize - 1; i++)          {              data[i] = data[i + 1];          }          currentSize--;      }      void display()      {          for (int i = 0; i < currentSize; i++)          {              std::cout << data[i] << " ";          }          std::cout << std::endl;      }      void reverse()      {          for (int i = 0; i < currentSize / 2; i++)          {              int temp = data[i];              data[i] = data[currentSize - i - 1];              data[currentSize - i - 1] = temp;          }      }      const int &operator[](int index) const      {          return data[index];      }      friend std::ostream &operator<<(std::ostream &os, const SList &list)      {          for (int i = 0; i < list.currentSize; i++)          {              os << list.data[i] << " ";          }          return os;      }      friend std::istream &operator>>(std::istream &is, SList &list)      {          for (int i = 0; i < list.maxSize; i++)          {              is >> list.data[i];          }          list.currentSize = list.maxSize;          return is;      }      const int size() const      {          return currentSize;      }      const int capacity() const      {          return maxSize;      }      const int find(int value) const      {          for (int i = 0; i < currentSize; i++)          {              if (data[i] == value)              {                  return i;              }          }          return -1;      }  };  int main()  {      SList list(10);                             // Create a list with size 10      std::cin >> list;                           // Input 10 integers      std::cout << list << std::endl;             // Display the list      list.reverse();                             // Reverse the list      list.remove(5);                             // Remove the 6th element      std::cout << list << std::endl;             // Display the list      list.insert(5, 100);                        // Insert 100 at the 6th position      std::cout << list << std::endl;             // Display the list      std::cout << list[5] << std::endl;          // Display the 6th element      std::cout << list.size() << std::endl;      // Display the size of the list      std::cout << list.capacity() << std::endl;  // Display the capacity of the list      std::cout << list.find(100) << std::endl;   // Find the index of 100      return 0;  }  1.drawio | | | | | |
| 1. **实验结果与分析**     主函数创建了一个容量为10的线性表，然后依次测试了从cin中输入数据、输出数据到cout、翻转、移除指定位置的元素、在指定位置插入某元素、下标索引、元素数量查询、容量查询、指定元素下标查询功能。 | | | | | |
| 1. **实验心得**   在这个实验中，我学会了运用面向对象的知识创建并维护一个线性表类，并且实现了线性表的插入删除、索引等必备功能。除此之外，我还使用了运算符重载实现了对istream和ostream类的输入输出支持。该实验锻炼了我的接口意识，强化了我对类的构造能力。 | | | | | |
| 1. **教师评语** | | | | | |
| 1. **实验成绩**   教师签名： 蒋振刚 批阅日期： 2025 年 3 月 30 日 | | | | | |

注：项目性质为 演示型、验证型、设计型、综合型和创新型。