Class CS\_UC211 Number 202115030121 Name Hao Yang

Machine number Lab2 Experiment date 2022/11/24 Report date 2022/11/26

**Experiment topic:**

Definition and Implementation of Linear List

1. **Overview**

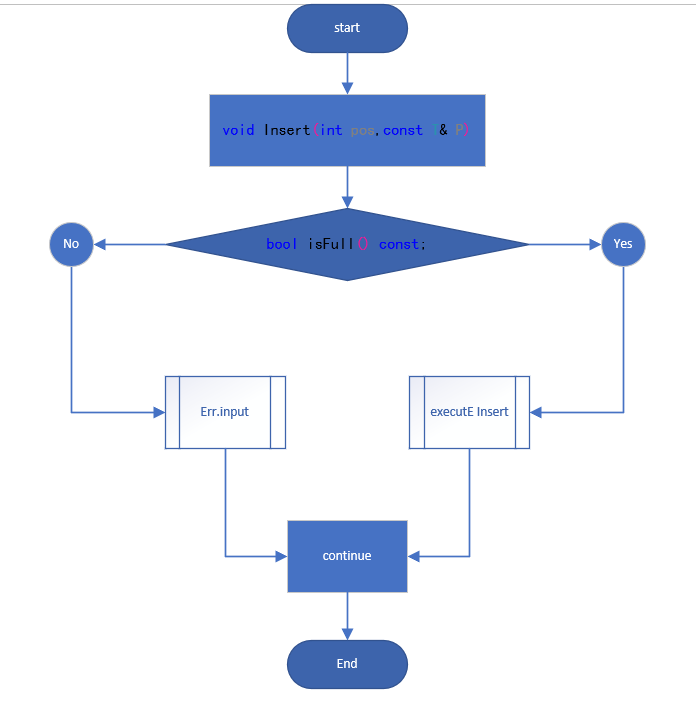
Completed all the experimental tasks, and constructed data to verify the correctness of the experiment.

**2. Experimental scheme**

2.1 Design scheme

Use C++ templates to increase the availability of data types. The form of a dynamic linear table is adopted.

2.2 Function call relation



2.3 Key algorithm implementation

* bool isFull() const;

Judgment\_ Length and\_ MAX\_ Whether LENGTH is equal in length. Equal means the array is full, otherwise, it means it is not full.

* void Insert(int pos,const T& P);

Determines whether the array is full. If it is full, it returns. Otherwise, it inserts. If pos is greater than\_ Length, assign pos as\_ length。 Press from\_ The pos of length decreases, and each element moves one bit backward to insert new elements into pos.

* void Delete(int pos);

From pos to\_ Length moves forward in increasing order.

* int getLocation(const T& P) const;

Press from 0 to\_ Length traverses sequentially. When the target element is found, its array subscript is returned.

* void Reverse();

Take\_ The intermediate value of length is traversed from 0 to its value, and the front and rear elements are exchanged in turn.

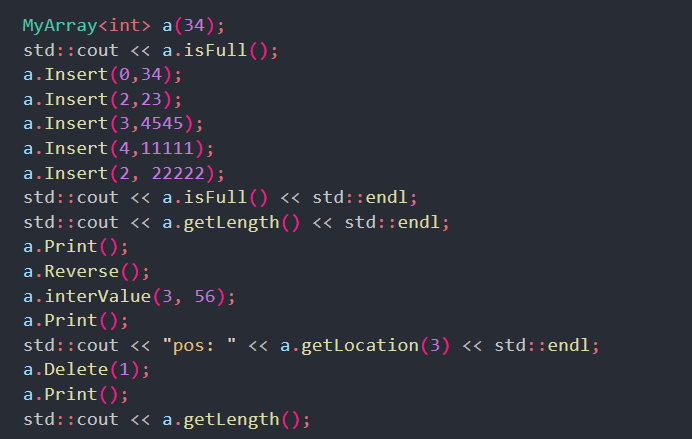
* void interValue(const T& Min, const T& Max);

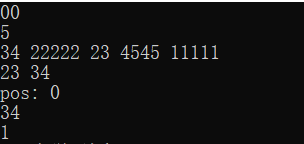
Create a new array, store the qualified elements in the new array, clear the old array space, and assign the new array to the old array.

**3. Experiment process**

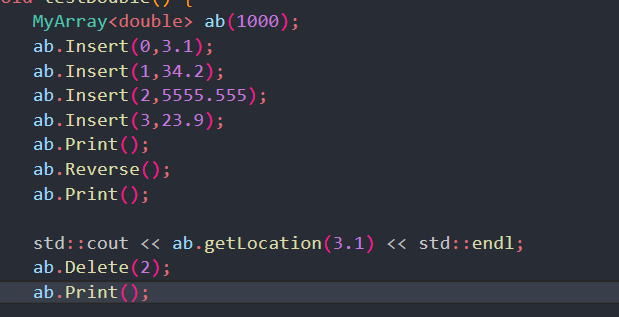
3.1 Test process

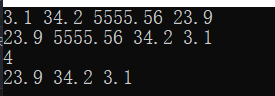
Integer type





Floating-point numbers





3.2 Debugging analysis

None.

**4. Evaluation analysis**

4.1 Analysis of experimental results

Through manual observation and machine inspection, the addition and deletion of the experiment are correct.

4.2 Algorithm performance evaluation

The time complexity to access any element in an array is O(1), a constant time. Inserting an element in the middle of an array requires moving all subsequent elements with a time complexity of O(n). Similarly, if you want to delete an element in the middle of an array, you need to move all the elements that follow, and the time complexity is O(n).

**5. Summary and experience**

Better understanding and use of C++ templates.

A deeper understanding of linear tables.

In future programming learning, you will be more sensitive to time complexity and design better programs.