

## Project Title: Template, Exception and File

### 1. Requirement Analysis

In this project, you try to use template to design a simple data structure, *sequence list*. It has the following features:

- The elements should be stored continuously, that is, the sequence should be stored in a continuous location (array). They should be ranged linearly one by one, e.g.  $a_1, a_2, a_3, \dots, a_{n-1}, a_n$ .
- Except for the first element (called head one) in the sequence, each element should have and only have a direct precursor element.
- Except for the last element (called tail one) in the sequence, each element should have and only have a direct following element.

The basic definition about the class Sequence List *SeqList* is as follows:

```
template <class ElemType> class SeqList {
    public:
        SeqList(int InitSize);    // Constructor.
        ~SeqList();              // Destructor.
        void Clear();            // Clear all the elements in the sequence list.
        bool IsEmpty()           // Return TRUE if the list is empty, otherwise return FALSE.
        int Length()             // Return the length of the list.
        ElemType Get(int i) const; // Return the value of the ith element.
        int Find(ElemType e) const; // Return element order if its value is equal to e,
                                    // return 0 if no such element exists.
        int Insert(int i, ElemType e); // Insert a new element with the value e in the ith
                                    // position. Alter the former ith element to the
                                    //  $(i+1)$ th
                                    // one. Return 1 if inserting is successful, otherwise
```

```

// return 0.

ElemType Delete(int i);    // Delete the ith element and return its value.

private:

ElemType *elem;    // Array to store the data of each element.

int length;    // Existing element number in the list, also the length of list.

int MaxSize;    // Maximum size of the array elem.

};

```

Write a Driver Program with the following requirements: firstly, put all the characters typed by user through keyboard into a sequence list; secondly, get out the first character of the list and put it into the middle; finally, display all the elements of the list on the screen. Note that there should be some Exception Handling Mechanisms in the program because of limitation of the storage capacity of sequence list. Prompt user that he / she inputs too many contents if the inputs exceed the storage. Test whether the exception handler runs normally.

All the outputs should be written into a file besides displayed on the screen during the running of the program. The file name could be assigned by the user while the program runs.

### Test cases:

#### Correct **input&output**

Please input the name of the output file: **seq.out**

Please input a sequence of characters (no more than 20):

**0123456789abab**

Display all the elements of the list:

**1234560789abab**

Any output other than this one is considered wrong.

## 2. Design Description

<pre>template &lt;class ElemType&gt; SeqList</pre>
<pre>-ElemType *elem -int length -int MaxSize</pre>
<pre>+SeqList(int InitSize) +~SeqList(); +void Clear(); +bool IsEmpty(); +int Length() +ElemType Get(int) +int Find(ElemType) +int Insert(int, ElemType) +ElemType Delete(int);</pre>

### 3. Debug Analysis

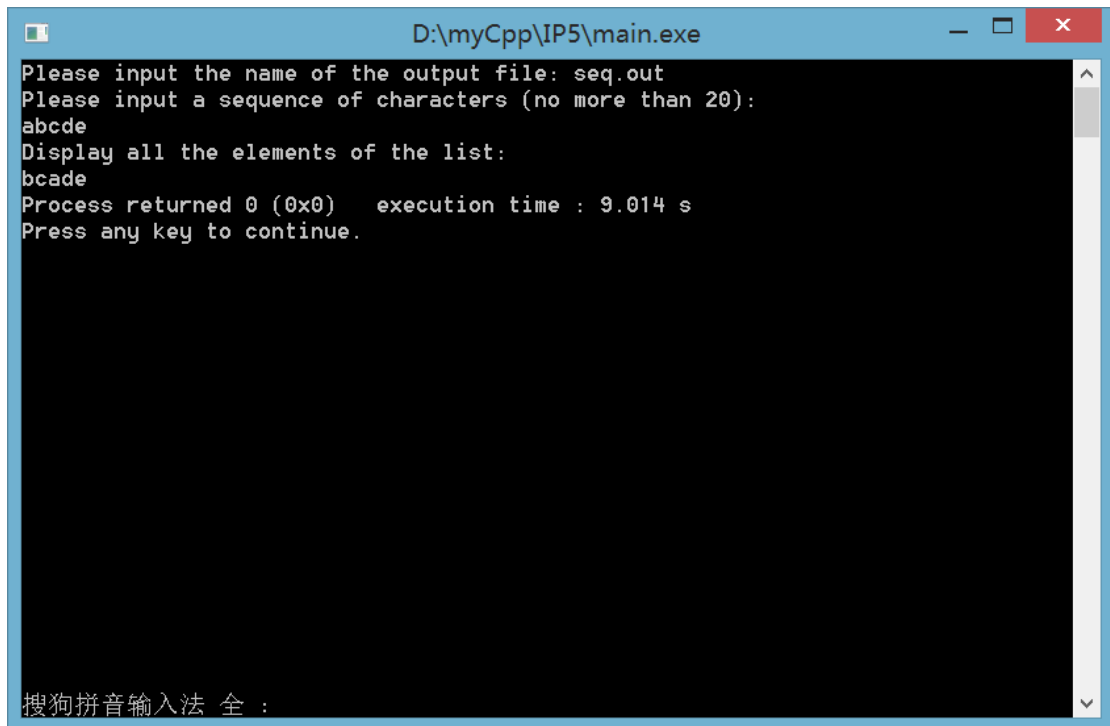
1. For the robustness and correctness of the program, troublesome conditions should be carefully taken into consideration. For instance, while prompting for input from the user, if the inputs exceed the storage, a corresponding exception should be thrown and handled.

2. While using templates, the type of variable should be carefully examined so that the function codes in the template class make perfect sense.

3. The code ‘template <class ElemType>’ should be added before each class function definition and the following class name should be modified with ‘<ElemType>’.

### 4. Test Result

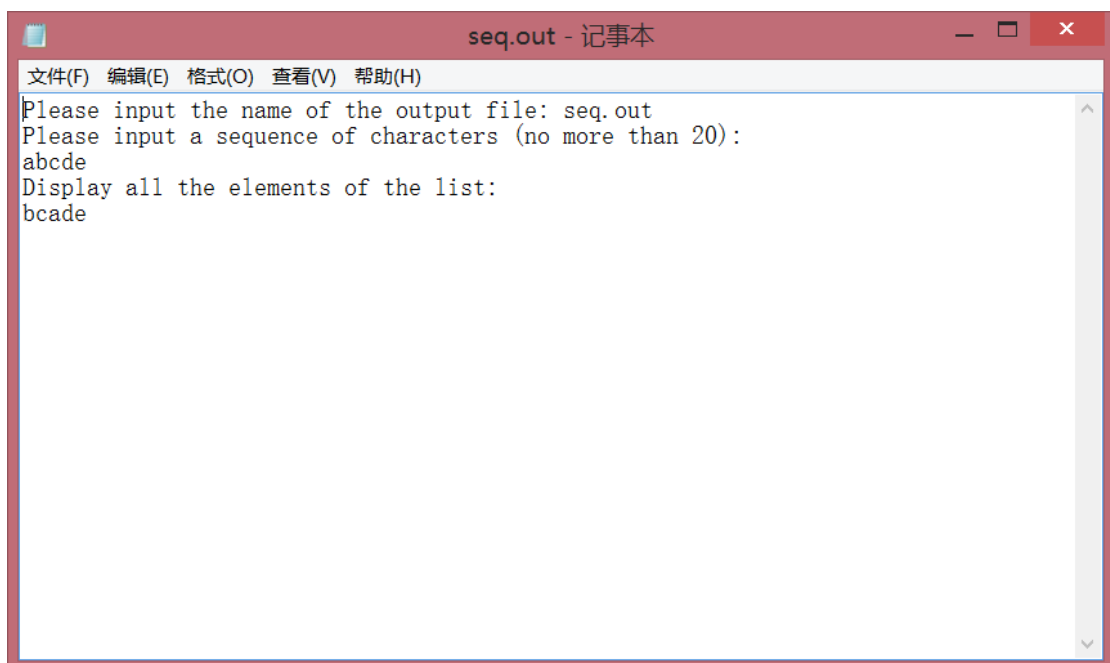
a )



```
D:\myCpp\IP5\main.exe
Please input the name of the output file: seq.out
Please input a sequence of characters (no more than 20):
abcde
Display all the elements of the list:
bcade
Process returned 0 (0x0)   execution time : 9.014 s
Press any key to continue.

搜狗拼音输入法 全 :
```

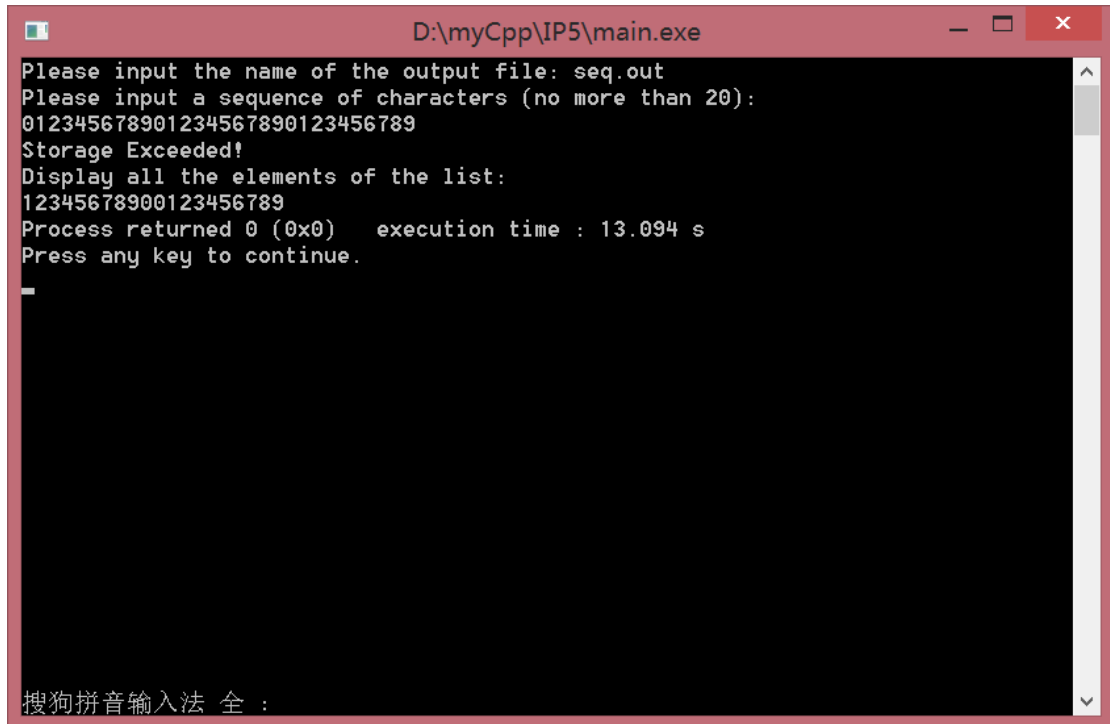
a\_1



```
seq.out - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
Please input the name of the output file: seq.out
Please input a sequence of characters (no more than 20):
abcde
Display all the elements of the list:
bcade
```

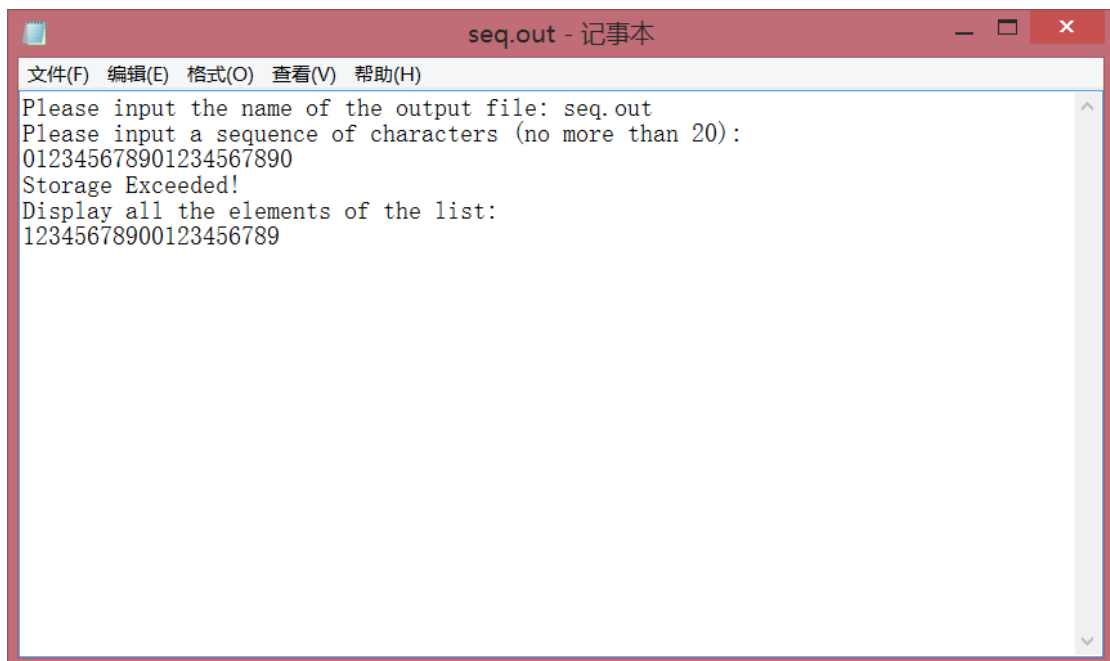
a\_2

b)



```
D:\myCpp\IP5\main.exe
Please input the name of the output file: seq.out
Please input a sequence of characters (no more than 20):
012345678901234567890123456789
Storage Exceeded!
Display all the elements of the list:
12345678900123456789
Process returned 0 (0x0)   execution time : 13.094 s
Press any key to continue.
-
搜狗拼音输入法 全 :
```

b\_1



```
seq.out - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
Please input the name of the output file: seq.out
Please input a sequence of characters (no more than 20):
012345678901234567890123456789
Storage Exceeded!
Display all the elements of the list:
12345678900123456789
```

b\_2

## 5. Appendix

The source code file is packed together with this report in the compressed files. However, the code as well as annotation is still provided in this report in case.

### [seqlist.h]

```
//File name:seqlist.h
//This program accomplishes the class SeqList and implements its member functions.
#ifndef _SEQUENCELIST_
#define _SEQUENCELIST_
using namespace std;
class NegativeSize{ };
class IndexOutOfRange{ };
class StorageExceeded{ };
template <class ElemType>
class SeqList
{
private:
    ElemType *elem; // Array to store the data of each element.
    int length; // Existing element number in the list, also the length of list.
    int MaxSize; // Maximum size of the array elem.
public:
    SeqList(int InitSize); // Constructor.
    ~SeqList(); // Destructor.
    void Clear(); // Clear all the elements in the sequence list.
    bool IsEmpty(){return (length==0);} // Return TRUE if the list is empty, otherwise
return FALSE.
    int Length(){return length;} // Return the length of the list.
    ElemType Get(int i) const; // Return the value of the ith element.
    int Find(ElemType e) const; // Return element order if its value is equal to e,
// return 0 if no such element exists.
    int Insert(int i, ElemType e); // Insert a new element with the value e in the ith
// position. Alter the former ith element to the (i+1)th
// one. Return 1 if inserting is successful, otherwise
// return 0.
    ElemType Delete(int i); // Delete the ith element and return its value.
};
template <class ElemType>
SeqList<ElemType>::SeqList(int InitSize) // Constructor.
```

```

{
    if (InitSize<=0) throw NegativeSize();
    MaxSize=InitSize;
    length=0;
    elem=new ElemType[MaxSize];
}
template <class ElemType>
SeqList<ElemType>::~SeqList() // Destructor.
{
    delete [] elem;
}
template <class ElemType>
void SeqList<ElemType>::Clear() // Clear all the elements in the sequence list.
{
    length=0;
}
template <class ElemType>
ElemType SeqList<ElemType>::Get(int i) const // Return the value of the ith element.
{
    if ((i<0)||i>=length)) throw IndexOutOfRange();
    return elem[i];
}
template <class ElemType>
int SeqList<ElemType>::Find(ElemType e) const // Return element order if its value is
equal to e, return 0 if no such element exists.
{
    for (int i=0;i<length;++i) if (e==elem[i]) return i;
    return 0;
}
template <class ElemType>
int SeqList<ElemType>::Insert(int i, ElemType e) // Insert a new element with the value e in
the ith
    // position. Alter the former ith element to the (i+1)th
    // one. Return 1 if inserting is successful, otherwise
    // return 0.
{
    if ((i<0)||i>length)) throw IndexOutOfRange();
    if (length==MaxSize) throw StorageExceeded();

```

```

        for (int j=length;j>i;--j) elem[j]=elem[j-1];
        elem[i]=e;
        ++length;
    }
template <class ElemType>
ElemType SeqList<ElemType>::Delete(int i) // Delete the ith element and return its value.
{
    if ((i<0)||i>=length) throw IndexOutOfRangeException();
    char t=elem[i];
    for (int j=i;j<length-1;++j) elem[j]=elem[j+1];
    --length;
    return t;
}
#endif

```

## [main.cpp]

```

#include "seqlist.h"
#include <iostream>
#include <fstream>
using namespace std;
int main()
{
    SeqList<char> list(20);
    cout<<"Please input the name of the output file: ";
    char fname[20],t;
    cin>>fname;
    ofstream out(fname);
    cout<<"Please input a sequence of characters (no more than 20):\n";
    cin.ignore(20,'\n');
    out<<"Please input the name of the output file: "<<fname<<endl<<"Please input a
sequence of characters (no more than 20):\n";
    int i=0;
    while (1){
        t=cin.get();
        ++i;
        out<<t;
        if (t=='\n') break;
        try

```



```

        {
            list.Insert(list.Length(),t);
        }
        catch (StorageExceeded)
        {
            cout<<"Storage Exceeded!\n";
            out<<"Storage Exceeded!\n";
            break;
        }
    }
    t=list.Delete(0);
    list.Insert(list.Length()/2,t);
    cout<<"Display all the elements of the list:\n";
    out<<"Display all the elements of the list:\n";
    for (i=0;i<list.Length();++i)
    {
        cout<<list.Get(i);
        out<<list.Get(i);
    }
    out.close();
}

```