Research Project 6:

Vector

Kuan Lu

Date: 2015-6-9

Chapter 1: Introduction

Given the declaration of class template Vetor below, write the bodies of the functions, and a main() to test all the facilities Vector provides.

Chapter 2: Coding Specification

I. Class and inheritance in this project:

```
                template
              class Vector
---------------------------------------------
public:
    Vector();           // create an empty vector
    Vector(int size);
    Vector(const Vector& r);// copy ctor
    virtual ~Vector();
    T& operator[](int index)
    int size();         // return the size of the vector
    int inflate(int addSize);
private:
    T *m_pElements;
    int m_nSize;
```

II. Source Code

(1) Vector.h

```
#ifndef __VECTOR_H__
#define __VECTOR_H__
#include<iostream>
#include<string>
using namespace std;

template <class T>
class Vector {
   public:
      Vector();
      Vector(int size);
       virtual ~Vector();
      Vector(const Vector& v);
       T& operator[](int);
       int size();
       int inflate(int addSize);
   private:
       T* m_elements;
       int m_size;
};

template <class T>
Vector<T>::Vector()
{
   m_elements= new T[20];
   m_size=20;
}

template <class T>
Vector<T>::Vector(int size):m_size(size)
{
   m_elements= new T[m_size];
}

template <class T>
Vector<T>::~Vector()
{
   delete m_elements;
}

template <class T>
Vector<T>::Vector(const Vector& v)
{
  int i;
  m_elements=new T[v.m_size];
  m_size=v.m_size;
  for(i=0;i<m_size;i++)
     m_elements[i]=v.m_elements[i];
}

template <class T>
T& Vector<T>::operator[](int index)
{
   if(index<m_size&&index>=0)
   {
      return m_elements[index];
   }
   else
    throw("IndexOutofBounds");
}

template <class T>
```

```
int Vector<T>::size()
{
    return m_size;
}

template <class T>
int Vector<T>::inflate(int addSize)
{
   T* tmp;
   tmp=m_elements;
   m_elements=new T[m_size+addSize];
   for(int i=0;i<m_size;i++)
      m_elements[i]=tmp[i];
   delete tmp;
   m_size+=addSize;
   return m_size;
}

#endif
```

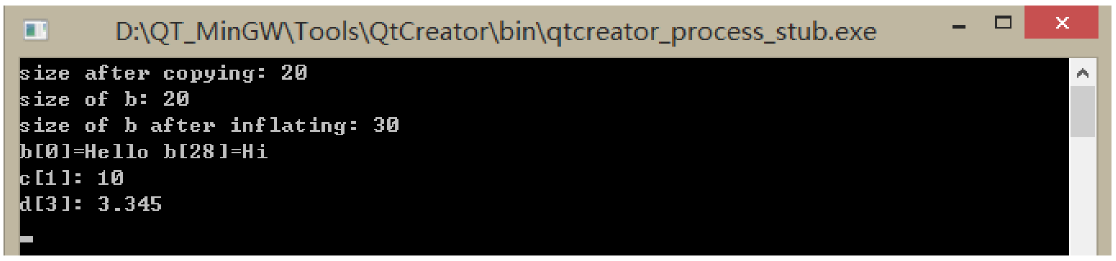
(2) Vector.cpp

```
#include "Vector.h"

void funct(Vector<string> a)
{
    cout<<"size after copying: "<<a.size()<<endl;
    return;
}

int main()
{
    Vector<string> a;                   //test Vector()
    Vector<string> b(20);                //test Vector(int size)
    a[0]="Hello";                       //test operator[](int index)
    b=a;                               //test copying one vector to another
    funct(b);
    cout<<"size of b: "<<b.size()<<endl; //test size()
    b.inflate(10);                      //test inflate(int addSize)
    cout<<"size of b after inflating: "<<b.size()<<endl;
    b[28]="Hi";
    cout<<"b[0]="<<b[0]<<" b[28]="<<b[28]<<endl;
    //output the result after inflating and copying
    Vector<int> c;
    Vector<double> d;
    c[1]=10;
    d[3]=3.345;
    cout<<"c[1]: "<<c[1]<<endl;
    cout<<"d[3]: "<<d[3]<<endl;
}
```

Chapter 3:   Test result



D:\QT_MinGW\Tools\QtCreator\bin\qtcreator_process_stub.exe

```
size after copying: 20
size of b: 20
size of b after inflating: 30
b[0]=Hello b[28]=Hi
c[1]: 10
d[3]: 3.345
```

Declaration

We hereby declare that all the work done in this project titled "Vector" is of my independent effort.