```
//Class: CSE 330
// Term: Spring 2014
// Instructor: George M. Georgiou
// Name: Seth Lemanek
// Lab 3
// Title: Vector.h
#ifndef VECTOR_H
#define VECTOR_H
#include <algorithm>
#include <assert.h>
using namespace std;

template <class T> class vector {
public:
    typedef T * iterator;
    typedef T value_type;
      //constructors
    vector  () { buffer = 0; resize(0); }
    vector  (unsigned int size) { buffer = 0; resize(size); }
    vector  (unsigned int size, T initial);
    vector  (vector & v);
    ~vector  () { delete [ ] buffer; }

     //member functions
    T       back () { return buffer [mySize -1];}
    iterator      begin() { return buffer; }
    int           capacity () { return myCapacity; }
    bool    empty () {return mySize == 0; }
    iterator      end () { return begin() + mySize; }
    T       front () { return buffer[0]; }
    void    pop_back () { mySize --; }
    void    push_back (T value);
    void    reserve (unsigned int newCapacity);
    void    resize (unsigned int newSize) { reserve(newSize); mySize = newSize; }
    int           size () { return mySize; }

    //operator
    T & operator [ ] (unsigned int index) { return buffer[index]; }

protected:
    unsigned int mySize;
    unsigned int myCapacity;
    T * buffer;
};

template <class T> vector<T>::vector (unsigned int size, T initial)
//creates vector with given size
//initialize all elements with given parameter
{
      buffer = 0;
      resize(size);
      fill (begin(), end(), initial);
}

template <class T> vector<T>::vector (vector & v)
//creates vector by copying from a previous one
{
      buffer = 0;
      resize(v.length());
      copy (v.begin(), v.end(), begin());
}

template <class T> void vector<T>::reserve (unsigned int newCapacity)
//reserves new capacity as large as the argument given
{
      if (buffer == 0)
        {
         mySize = 0;
         myCapacity = 0;
         }
      if (newCapacity <= myCapacity)
         return;
      T * newBuffer = new T [newCapacity];
      assert (newBuffer);
      copy (buffer, buffer + mySize, newBuffer);
      myCapacity = newCapacity;
      delete [ ] buffer;
      buffer = newBuffer;
}

template <class T> void vector<T>::push_back (T value)
//pushes a value to the back of the vector
{
      if (mySize >= myCapacity)
          reserve(myCapacity + 5);
      buffer [mySize++] = value;
}

#endif
```

```
//Class: CSE 330
//Term: Spring 2014
//Instructor: George M. Georgiou
//Name: Seth Lemanek
//Lab 3
//Title: Sieve of Eratosthenes
#include<iostream>
#include"vector.h"

using namespace std;

void sieve(vector<int> & values)
{
    unsigned int max = values.size();
    int i;

    // initalize all cells in the vector
    for (i = 0; i < max; i++)
      values[i] = i;
     //search for non-zero cells
    for (i = 2; i*i < max; i++)
    {
      if (values[i] != 0)
      {
          for (int j = i + i; j < max; j += i)
            values[j] = 0; //multiples of i have been cleared
      }
    }

}

int main()
{
    vector<int> nums (100);
    sieve(nums);//call sieve funct to make only prime numbers appear
    for (int i = 0; i < nums.size(); i++)
    {
         cout << nums[i] << " ";
    }
    cout << endl;
return 0;//end program with zero errors
}
```

```
Script started on Wed 23 Apr 2014 01:42:49 PM PDT
#]0;004470530@jb358-26:/students/csci/004470530##[?1034h[004470530@jb358-26
004470530]$ cd cse330/lab0##############g++ vector.h
sieve.cpp#######################[8Pcd cse330/lab0
bash: cd: cse330/lab0: No such file or directory
#]0;004470530@jb358-26:/students/csci/004470530#[004470530@jb358-26 004470530]$ cd
cse330/lab0##############g++ vector.h sieve.cpp#######################[8Pcd
cse330/lab03
#]0;004470530@jb358-26:/students/csci/004470530/cse330/lab03#[004470530@jb358-26
lab03]$ cd cse330/lab03##[K##############g++ vector.h sieve.cpp
#]0;004470530@jb358-26:/students/csci/004470530/cse330/lab03#[004470530@jb358-26
lab03]$ g++ vector.h sieve.cpp#######################[7Pcd
cse330/lab03##[K##############g++ vector.h sieve.cpp######################script -a
"CSE330_Lab3_Lemanek.txt
0 1 2 3 0 5 0 7 0 0 0 11 0 13 0 0 0 17 0 19 0 0 0 23 0 0 0 0 0 29 0 31 0 0 0 0 0 37
0 0 0 41 0 43 0 0 0 47 0 0 0 0 0 53 0 0 0 0 0 59 0 61 0 0 0 0 0 67 0 0 0 71 0 73 0
0 0 0 0 79 0 0 0 83 0 0 0 0 0 89 0 0 0 0 0 0 0 97 0 0
#]0;004470530@jb358-26:/students/csci/004470530/cse330/lab03#[004470530@jb358-26
lab03]$ exit
exit

Script done on Wed 23 Apr 2014 01:44:04 PM PDT
```