Московский государственный технический университет им. Н.Э. Баумана.

Факультет «Информатика и управление»

Кафедра ИУ5.
Курс «Программирование на основе классов и шаблонов»

Отчет по лабораторной работе №4
«Наследование. Множества»

Выполнил:		Проверил:
студент группы ИУ5-25Б		преподаватель каф. ИУ5
Петренко Сергей		
Подпись и дата:		Подпись и дата:

г. Москва, 2019 г.

Постановка задачи:

Разработать класс множество (MySet) на базе класса вектор (MyVector) для выполнения операций над множествами (+, -, *,+=,-=,*=,==) и функцию main( ) для его тестирования.

Класс вектор должен быть динамическим массивом, размер которого может автоматически изменяться (увеличиваться или уменьшаться) в процессе выполнения программы. Добавление элементов производится в конец вектора.

Для ускорения выполнения операций над множествами вектор, используемый классом множество, должен быть отсортирован (сортировку достаточно делать только при добавлении элемента в множество). Для поиска элементов множества следует использовать метод половинного деления.

Методы *add_element()* и *delete_element()* производного класса *MySet* перегружают одноименные методы базового класса *MyVector*, а остальные элементы класса *MyVector* наследуются классом *MySet* .

Разработка интерфейса класса:

Vector.h

```
#ifndef LAB4_MYVECTOR_H
#define LAB4_MYVECTOR_H
#include <iostream>
using namespace std;

const int MAX_SIZE=5;
class MyVector {
protected:
  int maxsize=5;
  void resize();
public:
  string* pdata = new string[maxsize];
  int size=0;
  MyVector(MyVector& v);
  MyVector(string el);
  MyVector();
  ~MyVector();
  void add_element(string el);
  void delete_element(int);
  int getSize();
  int getMaxSize();
  void sort();
  int find(string el);
  string operator[](int i);

  MyVector& operator=(MyVector& v){
     this->pdata=v.pdata;
     this->size=v.size;
     this->maxsize=v.maxsize;
     return *this;
  }

  friend ostream& operator<<(ostream& out, MyVector& v){
     if(v.size>0) {
        for (int i = 0; i < v.size; i++) {
           if(v[i]!="")
           out << v[i] << " ";
        }
     }
     else{
        cout<<"null";
     }
     return out;
  }

};
```

```
#endif //LAB4_MYVECTOR_H
```

MySet.h

```
#ifndef LAB4_MYSET_H
#define LAB4_MYSET_H
#include <iostream>
#include "string.h"
using namespace std;

class MySet: public MyVector  {
public:
  friend MySet operator + (MySet &s1, MySet& s2);
  friend MySet operator - (MySet &s1, MySet& s2);
  friend MySet operator*(MySet& s1,MySet& s2);
  MySet& operator-=(MySet&);
  MySet& operator*=(MySet&);
  MySet& operator+=(MySet&);
  bool operator==(MySet&);
  bool is_element(string);
  void delete_element(string);
  MySet();
  MySet(string el);
  MySet(const MySet& mySet);
  MySet& operator = (const MySet& s);

  friend ostream& operator<<(ostream& out, MySet s){
      s.sort();
      if(s.size>=0) {
          out <<"{ ";
          for (int i = 0; i < s.size; i++) {
              if(s[i]!="")
                  out << s[i];
          }
          out<<"} ";
      }
      else{
          cout<<"null";
      }
      return out;
  }
};
#endif //LAB4_MYSET_H
```

Текст программы:

MyVector.cpp

```
#include "MyVector.h"
using  namespace std;

   MyVector::MyVector(MyVector& v){
      this->size=v.size;
      this->maxsize=v.maxsize;
      this->pdata=v.pdata;
   }

   MyVector::MyVector(string el ){
      pdata[0]=el;
      size++;
   }

   MyVector::MyVector(){
   }

   MyVector::~MyVector(){
   }
string MyVector::operator[](int i) {
   return pdata[i]+" ";
}

   void MyVector:: delete_element(int k){
      if(size>0) {
         string *buf = new string[size - 1];
         if (k >= 0) {
         for (int i = k; i < size - 1; i++) {
            pdata[i] = pdata[i + 1];
         }
         size--;
         resize();
         for (int i = 0; i < size; i++) {
            buf[i] = pdata[i];
         }
         pdata = new string[size];
         for (int i = 0; i < size; i++) {
            pdata[i] = buf[i];
         }
         }
      }
   }
   void MyVector:: add_element(string el){
      size++;
      resize();
      string* buf = new string[size];
         for(int i=0;i<size-1;i++){
             buf[i]=pdata[i];
          }
      buf[size-1]=el;
      pdata = new string[maxsize];
         for(int i=0;i<size;i++){
            pdata[i]=buf[i];
         }
      }
```

```
  void MyVector::resize() {
     const int resizeValue=2;
      if(size>=maxsize){
        maxsize*= resizeValue;
     }
     else if(size<maxsize/2&&(maxsize/2)>MAX_SIZE){
         maxsize/=resizeValue;
     }
  }

  int MyVector::find(string el) {
     for(int i=0;i< size;i++){
        if(pdata[i]==el){
        return i;
        }
     }
     return -1;
  }

  int MyVector:: getSize(){
     return size;
  }
  int MyVector:: getMaxSize(){
     return maxsize;
  }
void MyVector::sort(){
  string o;
  int h;
  for (int j = size - 1; j > 0; j--) {
     h = 0;
     for (int i = 0; i < j; i++)
     {
       if(atoi(this->pdata[i].data())!=0&&atoi(this->pdata[i+1].data())!=0){
          if (std::atoi(this->pdata[i].data())> std::atoi(this->pdata[i+1].data()))
          {
             o = pdata[i];
             pdata[i] = pdata[i+1];
             pdata[i+1] = o;
             h++;
          }

       }
        else if ( this->pdata[i]> this->pdata[i + 1])
        {
           o = pdata[i];
           pdata[i] = pdata[i+1];
           pdata[i+1] = o;
           h++;
        }
     }
     if (h == 0)
     {
        for(int i=0;i<size-1;i++){
          if(pdata[i]==pdata[i+1]){
             this->delete_element(i);
          }
        }
        break;
     }
  }
}
```

MySet.cpp

```
#include "MySet.h"
MySet::MySet() {
}
MySet::MySet(string el){
this->pdata[0]=el;
this->size=1;
this->maxsize=5;
}
MySet::MySet(const MySet& mySet){
   this->pdata=mySet.pdata;
   this->maxsize=mySet.maxsize;
   this->size=mySet.size;
}
MySet& MySet::operator = (const MySet& s){
   this->size=s.size;
   this->pdata=s.pdata;
   this->maxsize=s.maxsize;
   return *this;
}
MySet operator + (MySet &s1, MySet& s2){
   MySet buf;
   for(int i=0;i<=s1.size;i++){
      buf.add_element(s1[i]);
   }
   for(int i=0;i<=s2.size;i++){
      if(!buf.is_element(s2[i])){
         buf.add_element(s2[i]);
      }
   }
   return buf;
}
```

```
MySet operator - (MySet &s1, MySet& s2){
  MySet buf;
  for(int i=0;i<s1.size;i++){
    if(!s2.is_element(s1[i])){
      buf.add_element(s1[i]);
    }
  }
  return buf;
}
MySet operator*(MySet& s1,MySet& s2){
  MySet buf;
  for(int i=0;i<s1.size;i++){
    if(s2.is_element(s1[i])){
      buf.add_element(s1[i]);
    }
  }
  return buf;
}
MySet& MySet::operator-=(MySet& s){
  for(int i=0;i<=s.size;i++){
    if(this->is_element(s[i])){
      this->delete_element(s[i]);
    }
  }
  return *this;
}
MySet& MySet::operator+=(MySet& s){
  for(int i=0;i<=s.size;i++){
    if(!this->is_element(s[i])){
      this->add_element(s[i]);
    }
  }
  return *this;
}
```

```
MySet& MySet::operator*=(MySet& s){
   MySet buf;
   for(int i=0;i<s.size;i++){
      if(this->is_element(s[i])){
         buf.add_element(s[i]);
      }   }
   *this = buf;
  return *this;
}
bool MySet::operator==(MySet& s){
   int res =0;
   if(this->size==s.size){
      for(int i = 0; i<s.size;i++){
         if(this->is_element(s[i])){
            res++;
         } }
      if(res==size)return true;  }   else return false;
}
void MySet::delete_element(string el){
   string st = el.substr(0, el.size()-1);
   int a = find(st);
   if(a!=-1){
      string *buf = new string[size - 1];
      for (int i = a; i < size - 1; i++) {
         pdata[i] = pdata[i + 1];
      } size--;
      resize();
      for (int i = 0; i < size; i++) {
         buf[i] = pdata[i];   }
      pdata = new string[size];
      for (int i = 0; i < size; i++) {
         pdata[i] = buf[i];  }
   }
   sort();
```

```
bool MySet:: is_element(string el){
   string st = el.substr(0, el.size()-1);
   for(int i=0;i<=size;i++){
      if(pdata[i]==st){
         return true;
      }
   }
   return false;
}
```

Анализ данных:

```
/Users/sergei/CLionProjects/Lab4/cmake-build-debug/Lab4
MaxSize 5
Вектор v: Hello!  Привет!  Привет!  Привет!  Привет!  Привет!
MaxSize 10
Вектор v: Hello!  Привет!  Привет!  Привет!  Привет!  Привет!  Привет!  Привет!  Привет!
Вектор v1: Hello!  Привет!  Привет!  Привет!  Привет!  Привет!  Привет!  Привет!  Привет!
Вектор v1: null
Множество s: { Hello! No Yes! Привет! }
Множество s1: { Cat No Привет! }
Множество s2=s1-s: { Cat  }
Множество s1: { Cat No Привет! }
Множество s: { Hello! No Yes! Привет! }
Множество s2=s-s1: { Hello!  Yes!  }
Множество s1: { Cat No Привет! }
Множество s: { Hello! No Yes! Привет! }
Множество s2=s1+s: {  Cat  Hello!  No  Yes!  Привет!  }
Множество s1: { Cat No Привет! }
Множество s: { Hello! No Yes! Привет! }
Множество s2=s1*s: { No  Привет!  }
Множество s1: { Cat No Привет! }
Множество s: { Hello! No Yes! Привет! }
Множество s3=s2: { No  Привет!  }
Множество s3=s2
Множество s3!=s1
Множество s1!=s3
```

```
Введите элемент 1 множества: 1
Введите элемент 1 множества: 2
Введите элемент 1 множества: 3
Введите элемент 1 множества: 4
Введите элемент 1 множества: 5
Введите элемент 1 множества: 6
Введите элемент 1 множества: 7
Введите элемент 1 множества: 8
Введите элемент 1 множества: 9
Введите элемент 1 множества: 10
Введите элемент 1 множества: stop
Введите элемент 2 множества: 11
Введите элемент 2 множества: 12
Введите элемент 2 множества: 13
Введите элемент 2 множества: 14
Введите элемент 2 множества: 15
Введите элемент 2 множества: 3
Введите элемент 2 множества: stop
{ 1 2 3 4 5 6 7 8 9 10 }
{ 3 11 12 13 14 15 }
+ {  1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  }
- [1]{ 1  2  4  5  6  7  8  9  10  }
- [2]{ 11  12  13  14  15  }
* { 3  }
```

```
Введите элемент 1 множества: 1
Введите элемент 1 множества: 2
Введите элемент 1 множества: 3
Введите элемент 1 множества: 4
Введите элемент 1 множества: 5
Введите элемент 1 множества: 6
Введите элемент 1 множества: 7
Введите элемент 1 множества: 8
Введите элемент 1 множества: 9
Введите элемент 1 множества: 10
Введите элемент 1 множества: stop
Введите элемент 2 множества: 11
Введите элемент 2 множества: 12
Введите элемент 2 множества: 13
Введите элемент 2 множества: 14
Введите элемент 2 множества: 15
Введите элемент 2 множества: 3
Введите элемент 2 множества: stop
{ 1 2 3 4 5 6 7 8 9 10 }
{ 3 11 12 13 14 15 }
+ {   1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  }
- [1]{ 1  2  4  5  6  7  8  9  10  }
- [2]{ 11  12  13  14  15  }
* { 3  }
```