

C++作业

罗悦 2016220304022

习题 9.7:

请读者根据本小节描述的思路,完成所有的类设计并编码测试。提示:

可以为 calculator 类设计新的接口函数来设置选定的输入/输出设备。

所编写的程序如下所示:

```
#include <iostream>
#include <string>
using namespace std;

class device
{
    public:
        virtual void work()const=0;

    private:
};

class inputdev:public device{
    public:
        void work();
};

class outputdev:public device{
    public:
        void work();
};

class key-board:public inputdev{
    public:
        void work();
        string input();
};
```

```

class monitor:public outputdev{
    public:
        void work();
        void display(string content);
};

class calculator{
    private:
        string calculate(string expression);
    public:
        void work();
};

int main()
{
    return 0;
}

```

习题 10.1：函数 `int compare(a, b)` 用于比较两个数的大小，它返回一个整型值：1 表示 $a > b$, 0 表示 $a == b$, -1 表示 $a < b$ 。请将这个函数编写成模板形式，然后用各种不同的类型（包括第九章设计的形体类型）来测试这个函数。

所编写的程序如下所示：

```

#include <iostream>
#include <string>
#include <cmath>
using namespace std;

template<typename T>
int compare(const T& a,const T& b){
    if(a>b){
        return 1;
    }
    else if(a==b){
        return 0;
    }
    else{
        return -1;
    }
}

```

```

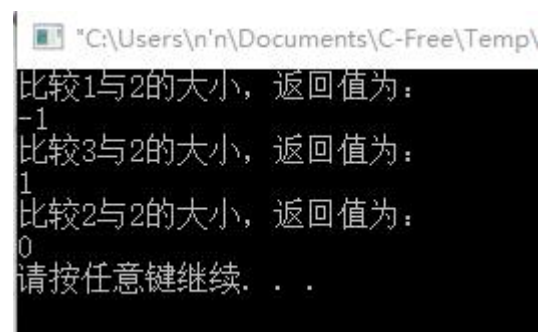
    }
    else if(a<b){
        return -1;
    }
}

int main()
{
    cout<<"比较 1 与 2 的大小, 返回值为: "<<endl;
    cout<<compare(1,2)<<endl;
    cout<<"比较 3 与 2 的大小, 返回值为: "<<endl;
    cout<<compare(3,2)<<endl;
    cout<<"比较 2 与 2 的大小, 返回值为: "<<endl;
    cout<<compare(2,2)<<endl;

    return 0;
}

```

测试结果如下图所示：



```

"C:\Users\n'n\Documents\C-Free\Temp\
比较1与2的大小, 返回值为:
-1
比较3与2的大小, 返回值为:
1
比较2与2的大小, 返回值为:
0
请按任意键继续. . .

```

运算结果和实验结果如上图所示

习题 10.2：请为函数 `compare()` 重载一个非模板形式和一个特化。

所编写程序如下所示：

```

#include <iostream>
#include <string>
#include <cmath>
using namespace std;

```

```

template<typename T>
int compare(const T& a,const T& b){
    if(a>b){
        return 1;
    }
    else if(a==b){
        return 0;
    }
    else if(a<b){
        return -1;
    }
}

template<typename T,typename U>
int compare(const T& a,const U& b){
    if(a>b){
        return 1;
    }
    else if(a==b){
        return 0;
    }
    else if(a<b){
        return -1;
    }
}

int main()
{
    cout<<"比较 1 与 2 的大小, 返回值为: "<<endl;
    cout<<compare(1,2)<<endl;
    cout<<"比较 3 与 2 的大小, 返回值为: "<<endl;
    cout<<compare(3,2)<<endl;
    cout<<"比较 2 与 2 的大小, 返回值为: "<<endl;
    cout<<compare(2,2)<<endl;
    cout<<"比较 3.1 与 2 的大小, 返回值为: "<<endl;
    cout<<compare(3.1,2)<<endl;
    cout<<"比较 3 与 4.5 的大小, 返回值为: "<<endl;
    cout<<compare(3,4.5)<<endl;
    cout<<"比较 3.1 与 3.10 的大小, 返回值为: "<<endl;
    cout<<compare(3.1,3.10)<<endl;

    return 0;
}

```

```
}v
```

如上程序所示，新加了一个非模板函数和一个特化函数，测试结果如下图所示：



```
"C:\Users\n\n\Documents\C-Free\Temp\未命名35.exe"  
比较1与2的大小，返回值为：  
-1  
比较3与2的大小，返回值为：  
1  
比较2与2的大小，返回值为：  
0  
比较3.1与2的大小，返回值为：  
1  
比较3与4.5的大小，返回值为：  
-1  
比较3.1与3.10的大小，返回值为：  
0  
比较' happy day'与'hello'的大小，返回值为：  
-1  
请按任意键继续. . .
```

习题 10.8：请为向量类编写两个泛型只读算法。

编写程序如下所示：

```
v_int vec = {1,2,3,4,5,6};  
v_str vec_str1 {"A","B","C","D"};  
//accumulate: 元素求和，前两个参数指定求和元素的范围，第三个参数是和的初  
值。  
auto sum_int = accumulate(vec.cbegin(),vec.cend(),0);  
cout << sum_int << endl;  
auto sum_str = accumulate(vec_str1.cbegin(),vec_str1.cend(),string("")); //string 初始  
化为空串。  
cout << sum_str << endl;  
l_char lst_char {"A","B","C","D"};  
//equal: 用于确定两个序列是否保存相同的值，返回布尔值。第二个序列至少应  
该和第一个序列一样长。  
cout << equal(vec_str1.cbegin(),vec_str1.cend(),lst_char.cbegin()) << endl;  
fill(vec.begin(),vec.begin() + vec.size()/2,0); //fill: 接受一对迭代器表示范围，用指  
定值替代范围内的元素。  
fill_n(vec.begin()+3,3,1); //fill_n: 用指定值替代从指定元素开始的多个元素。  
for (const auto s : vec)
```

```

{
    cout << s << " ";
}
cout << endl;

//back_inserter:通过插入迭代器将元素添加到容器中。
v_int vec2;
auto it = back_inserter(vec2);
*it = 10;
fill_n(back_inserter(vec2),5,0);//向 vec2 的末尾添加五个元素。
for (const auto s : vec2)
{
    cout << s << " ";
}
cout << endl;

//copy: 此算法将输入范围中的元素拷贝到新的序列中，三个参数都是迭代器，
返回值为尾迭代器。
auto ret = copy(begin(vec),end(vec),begin(vec2));
for (const auto s : vec2)
{
    cout << s << " ";
}
cout << endl;

//replace: 将序列中某个元素全部替换为指定的元素（最后一个参数）。
replace(begin(vec2),end(vec2),0,1);
for (const auto s : vec2)
{
    cout << s << " ";
}
cout << endl;

//replace_copy: 原序列不变，新序列是原序列的拷贝，只是替换了原序列中某个
元素为新的元素。
replace_copy(begin(vec2),end(vec2),back_inserter(vec),1,0);
for (const auto s : vec)
{
    cout << s << " ";
}
cout << endl;

v_int vec3 = {1,2,1,6,3,4,4,5};
//sort: 默认使用升序排序，如果要使用降序则用反向迭代器。

```

```

sort(vec3.begin(),vec3.end());
for (const auto s : vec3)
{
    cout << s << " ";
}
cout << endl;
//unique: 重排输入范围，使得每个元素只出现一次，返回指向不重复区域之后
一个位置的迭代器。
auto end_unique = unique(vec3.begin(),vec3.end());
vec3.erase(end_unique,vec3.end());//删除重复元素。
for (const auto s : vec3)
{
    cout << s << " ";
}
cout << endl;

```

实验结果如下图所示：

```

21
ABCD
1
0 0 0 1 1 1
10 0 0 0 0 0
0 0 0 1 1 1
1 1 1 1 1 1
0 0 0 1 1 1 0 0 0 0 0
6 5 4 4 3 2 1 1
6 5 4 3 2 1

```

习题 针对 p266 例 10-6，不用标准库，自行编写泛型编程算法实现 map 模板，同时实现例 10-6 相同的功能。

编写程序如下所示：

```

#include<iostream>
#include<string>
using namespace std;
template<typename T>
class test{
private:
    int num;

```

```

public:
    T* key;
    T* val;
    T look(T word){
        int j;
        for(j = 0;j < num;j++){
            if(key[j] == word)return val[j];
        }
        return string("未录入");
    }
    test(T array1[],T array2[]){
        cout<<sizeof(array1)<<endl;
        cout<<sizeof(array1[0])<<endl;
        num = sizeof(array1)/sizeof(*array1)+1;
        key = (T*)malloc(sizeof(T)*num);
        val = (T*)malloc(sizeof(T)*num);
        cout<<num<<endl;
        int i;
        for(i = 0;i < num;i++){
            key[i] = array1[i];
            val[i] = array2[i];
            cout<<key[i]<<endl;
            cout<<val[i]<<endl;
        }
    }
};

int main(){
    string a[] = {
        "zoo",
        "mammal",
        "felid",
        "tiger",
        "lynx"
    };
    string b[] = {
        "动物园",
        "哺乳动物",
        "猫科动物",
        "老虎",
        "猞猁"
    };
    test <string> test(a,b);
}

```



```
    return 0;  
}
```

测试结果如下图所示：

