Final Project: Simple Circuit

实验报告

17343107 王明业

一、 代码运行

1. 集成开发工具（IDE）：Xcode Version 9.2 (9C40b)
2. 头文件：

```
#include <iostream>
#include <string>
#include <vector>
#include <map>
```

#include <iostream>

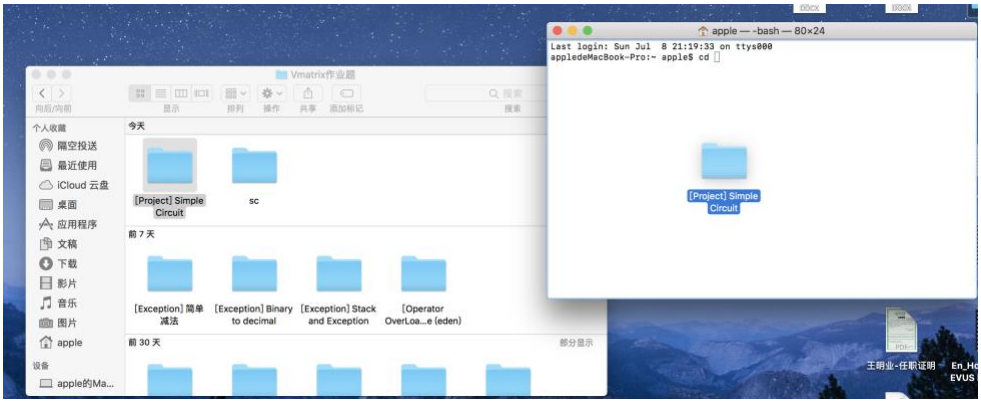
#include <string>

#include <vector>

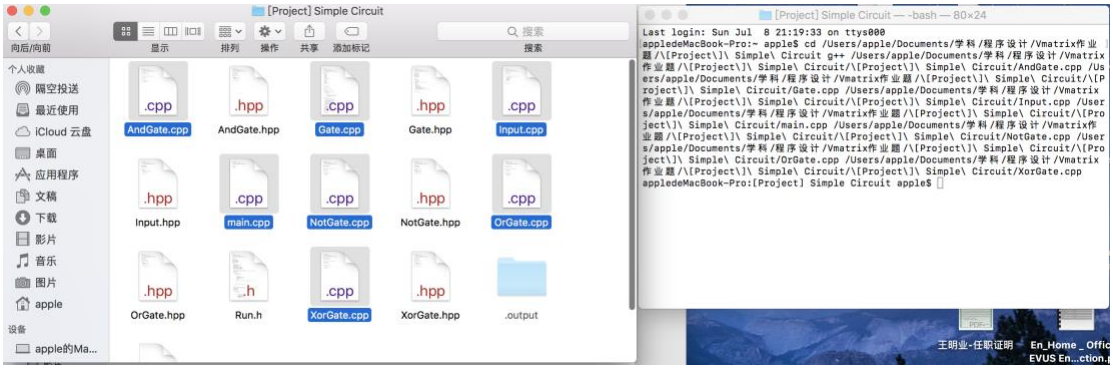
#include <map>

3. g++指令编译与运行：

I. 打开 macOS 系统自带的终端（Terminal），输入”cd “，然后输入源代码所在目录（可通过将源代码所在文件夹用鼠标拖入终端窗口实现）。



II. 然后输入“g++ ”后跟所有源代码中.cpp 文件的文件名，可用空格分隔，完成后回车。此后，源代码所在文件夹会出现 a.out 文件。其中的.o 文件，即对象文件（object file），内含汇编码，是由源码向机器码的过渡。



III. 然后在终端输入“./a.out”，完成后回车。此时，程序会在终端内运行。

二、 使用步骤

1. 向程序输入目标电路的输入

```
Please Enter the Input:
(Format:'name' 'value'; Example:A 1; Input "END" to finish)
```

2. 向程序输入目标电路所需的逻辑门

```
Please Enter the Gate:
(Format:'type' 'name'; Example:AndGate 1; Input "END" to finish)
```

3. 向程序输入对目标电路的描述

```
Please describe the Circuit:
(Format:'type' 'name' to 'type' 'name'; Eaxmple: Input A to AndGate 1; Input "END"
to finish)
```

4. 示例输入

实现全加器：（现默认输入 0，0，1，可修改）

A 0

B 0

C 1

END

AndGate 1

AndGate 2

AndGate 3

XorGate 1

XorGate 2

OrGate 1

OrGate 2

END

Input A to XorGate 1

Input B to XorGate 1

XorGate 1 to XorGate 2

Input C to XorGate 2

Input A to AndGate 1

Input B to AndGate 1

Input A to AndGate 2

Input C to AndGate 2

Input B to AndGate 3

Input C to AndGate 3

AndGate 1 to OrGate 1

AndGate 2 to OrGate 1

OrGate 1 to OrGate 2

AndGate 3 to OrGate 2

END

XorGate 2

OrGate 2

END

输出：

```
Output is/are 1 0
```

符合逻辑

三、 设计思路

1. 封装：

   将要素分为：门（Gate）、输入（Input）、与门（AndGate）、或门（OrGate）、异或门（XorGate）、非门（NotGate）、运行（Run），分别用类封装以上要素。

2. 运算符重载的应用：

   I. 拷贝构造函数

```
void Gate::operator= (Gate& another)
{
    this->name=another.name;
    this->graph=another.graph;
    this->output=another.output;
    this->connection=another.connection;
    this->in[0]=another.in[0];
    this->in[1]=another.in[1];
    this->output=another.output;
    this->out=another.out;
}
```

II. 门的输出：

```
ostream& operator<< (ostream& out, Gate* p)
{
    out<<p->theGraph();
    return out;
}
```

3. 继承：

输入（Input）、与门（AndGate）、或门（OrGate）、异或门（XorGate）、非门（NotGate）均继承自父类：门（Gate）

```
class Input: public Gate
{

class AndGate: public Gate


class OrGate: public Gate
{

class XorGate: public Gate
{

class NotGate: public Gate
{
```

4. 多态

```
virtual void compute()=0;
```

使用虚函数实现各类门的逻辑算法，例如与门和或门：

```
void AndGate::compute()
{
    out=in[0]*in[1];
}

void OrGate::compute()
{
    out=in[0]+in[1];
    if(out>1)
        out=1;
}
```

5. 异常

Ⅰ. 当 Input 的输入不合法时：

```
if(value!=1&&value!=0)
    throw value;
```

```
try
{
    cin >> value;
    Input ipt(InputName, value);
    input[iptNum] = ipt;
    iptNum++;
    inputMap[InputName] = ipt;
}
catch (int)
{
    cout << "Error: The Value of an Input must be 0 or 1." << endl;
    exit(-1);
}
```

Ⅱ. 当门的输入不合法时：

```
else
    throw GateType;
```

```
catch (string)
{
    cout << "Error: This type of Gate does not exsit. " << endl;
    exit(-1);
}
```

（这里的 try 和 catch 间有许多代码，仅截取 throw 部分和 catch 部分作为示例）

Ⅲ. 当电路连接不合法时：

```
if(infull==true)
    throw infull;
```

```
catch (bool)
{
    cout << "Connect Error. " << endl;
    exit(-1);
}
```

（这里的 try 和 catch 间有许多代码，仅截取 throw 部分和 catch 部分作为示例）