面向对象程序设计C++ 第5次实验

题目1

实现思路

Point类

变量：两个坐标x,y分别保存其平面直角坐标系中的横纵坐标

方法：

getX()和getY()分别返回其x,y值

r()使用sqrt方法计算其相对源点距离

theta使用atan方法计算其相对x轴正方向的夹角(theta=arctan(y/x))，其中需要对x=0和x<0的情况进行特殊处理，使用acos(-1.0)获得圆周率值

distance同样使用sqrt方法计算其相对另一点的距离

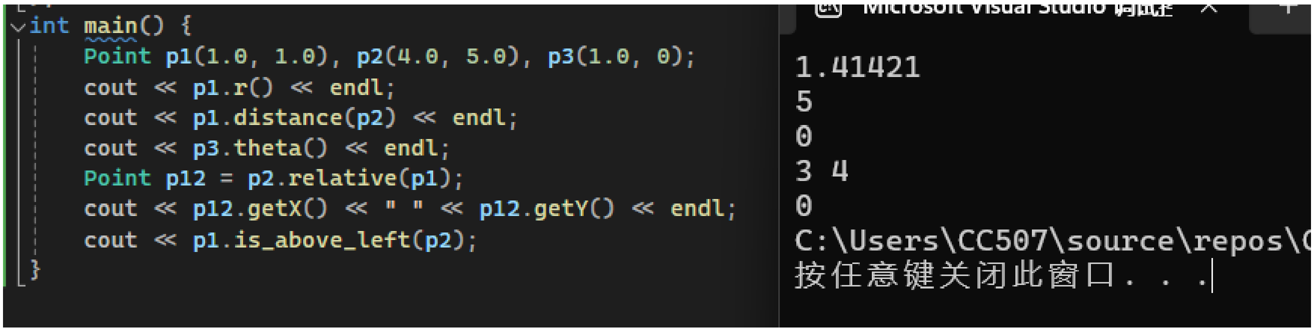
relative横纵坐标分别相减即可得到相对坐标

is_above_left对比坐标大小即可

源码

```
#include<iostream>
using namespace std;
class Point
{
public:
    Point() {
        x = 0;
        y = 0;
    }
    Point(double x,double y) {
        this->x = x;
        this->y = y;
    }
    double getX(){
        return x;
    }
    double getY() {
        return y;
    }
    double r() {
        return sqrt(x * x + y * y);
    }
    double theta() {
        if (x == 0) {
            if (y > 0)return acos(-1.0) / 2;
            else if (y == 0)return 0;
            else return -acos(-1.0) / 2;
        }
        else if(x > 0) {
            return atan(y / x);
        }
        else {
            if (y == 0)return 0;
            else if (y > 0)return atan(y / x) + acos(-1.0);
            else return atan(y / x) - acos(-1.0);
        }
    }
    double distance(const Point& p) {
        return sqrt((p.x - x) * (p.x - x) + (p.y - y) * (p.y - y));
    }
    Point relative(const Point& p) {
        Point r(x - p.x, y - p.y);
        return r;
    }
    bool is_above_left(const Point& p) {
        return (x < p.x) && (y > p.y);
    }
private:
    double x, y;
};
int main() {
    Point p1(1.0, 1.0), p2(4.0, 5.0), p3(1.0, 0);
    cout << p1.r() << endl;
    cout << p1.distance(p2) << endl;
    cout << p3.theta() << endl;
    Point p12 = p2.relative(p1);
    cout << p12.getX() << " " << p12.getY() << endl;
    cout << p1.is_above_left(p2);
}
```

运行结果



```
int main() {
    Point p1(1.0, 1.0), p2(4.0, 5.0), p3(1.0, 0);
    cout << p1.r() << endl;
    cout << p1.distance(p2) << endl;
    cout << p3.theta() << endl;
    Point p12 = p2.relative(p1);
    cout << p12.getX() << " " << p12.getY() << endl;
    cout << p1.is_above_left(p2);
}
```

```
1.41421
5
0
3 4
0
C:\Users\CC507\source\repos\(
按任意键关闭此窗口. . .
```

题目2

实现思路

is_substring(const char* p)方法：遍历字符串的每个位置，通过strncmp逐一比对。

is_substring(const String& s)方法：直接调用is_substring(s.str)。

substring(int start, int length)方法：创建一个新的char*指针，使用strncpy复制过去，在末尾添加'\0'。

find_replace_str(const char* find_str, const char* replace_str)方法：先遍历一遍原字符串，获得成功匹配find_str子串的次数count，再创建通过这个count创建适当大小的newstr；重新遍历原字符串，若匹配find_str则为新字符串添加replace_str，若不匹配则直接复制原字符串的字符。

remove_spaces()方法：调用find_replace_str将空格替换为空字符串。

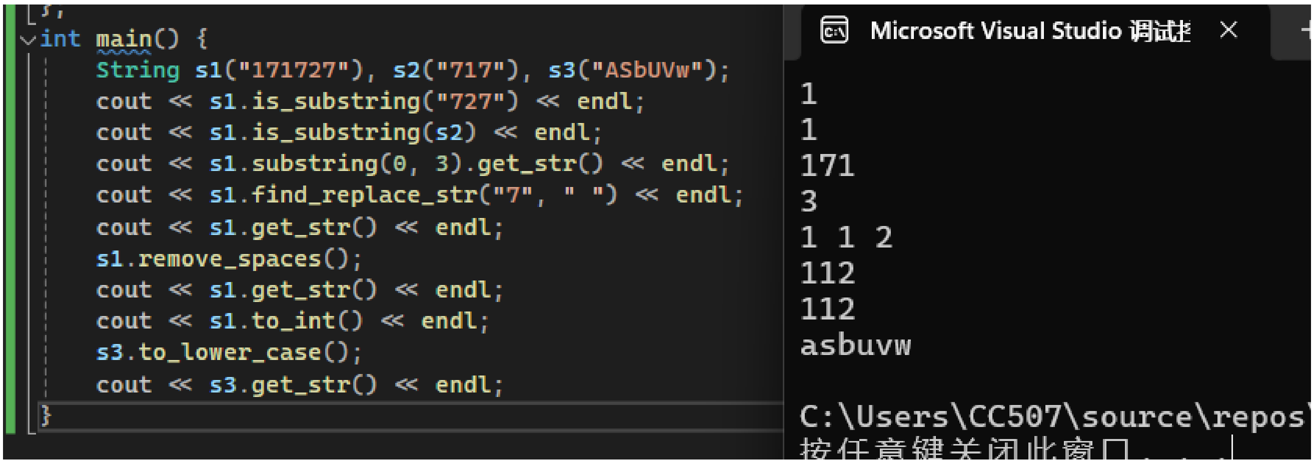
to_int()方法：遍历字符串，对每个字符减去'0'后*10再加下一个字符。

to_lower_case()方法：遍历字符串，对每个大写的字符转为小写。

源码

```
#include<cstring>
#include<cstdlib>
#include<iostream>
using namespace std;
class String {
    char* str;
public:
    String() {
        str = NULL;
    }
    String(const char* p) {
        str = new char[strlen(p) + 1];
        strcpy(str, p);
    }
    ~String() {
        delete[]str;
        str = NULL;
    }
    int length() {
        return strlen(str);
    }
    char* get_str() {
        return str;
    }
    char& char_at(int i) {
        if (i < 0 || i >= strlen(str)) {
            cerr << "超出字符串范围!\n";
            exit(-1);
        }
        return str[i];
    }
    bool is_substring(const char* p) {
        for (int i = 0; i <= strlen(str) - strlen(p); i++) {
            if (strncmp(p, str + i, strlen(p)) == 0)return true;
        }
        return false;
    }
    bool is_substring(const String& s) {
        return is_substring(s.str);
    }
    String substring(int start, int length) {
        char* s = new char[length + 1];
        strncpy(s, str + start, length);
        s[length] = '\0';
        String sstr(s);
        delete[] s;
        return sstr;
    }
    int find_replace_str(const char* find_str, const char* replace_str) {
        int count = 0;
        char* p = str;
        while (*p != '\0') {
            if (strncmp(p, find_str, strlen(find_str)) == 0) {
                count++;
                p += strlen(find_str);
            }
            else p++;
        }
        if (count == 0)return 0;
        char* newstr = new char[strlen(str) - count * strlen(find_str) + count * strlen(replace_str) + 1];
        p = str;
        char* q = newstr;
        while (*p != '\0') {
            if (strncmp(p, find_str, strlen(find_str)) == 0) {
                strcpy(q, replace_str);
                q += strlen(replace_str);
                p += strlen(find_str);
            }
            else {
                *q = *p;
                p++;
                q++;
            }
        }
        delete[]str;
        str = newstr;
        return count;
    }
    void remove_spaces() {
        find_replace_str(" ", "");
    }
    int to_int() {
        int count = 0;
        for (int i = 0; i < strlen(str); i++) {
            count *= 10;
            count += str[i] - '0';
        }
        return count;
    }
    void to_lower_case() {
        for (int i = 0; i < strlen(str); i++) {
            if (str[i] >= 'A' && str[i] <= 'Z') {
                str[i] -= 'A' - 'a';
            }
        }
    }
};
int main() {
    String s1("171727"), s2("717"), s3("ASbUVw");
    cout << s1.is_substring("727") << endl;
    cout << s1.is_substring(s2) << endl;
    cout << s1.substring(0, 3).get_str() << endl;
    cout << s1.find_replace_str("7", " ") << endl;
    cout << s1.get_str() << endl;
    s1.remove_spaces();
    cout << s1.get_str() << endl;
    cout << s1.to_int() << endl;
    s3.to_lower_case();
    cout << s3.get_str() << endl;
}
```

运行结果



```
int main() {
    String s1("171727"), s2("717"), s3("ASbUVw");
    cout << s1.is_substring("727") << endl;
    cout << s1.is_substring(s2) << endl;
    cout << s1.substring(0, 3).get_str() << endl;
    cout << s1.find_replace_str("7", " ") << endl;
    cout << s1.get_str() << endl;
    s1.remove_spaces();
    cout << s1.get_str() << endl;
    cout << s1.to_int() << endl;
    s3.to_lower_case();
    cout << s3.get_str() << endl;
}
```

```
1
1
171
3
1 1 2
112
112
asbuvw

C:\Users\CC507\source\repos
按任意键关闭此窗口. . .
```

题目3

实现思路

使用vector<vector>存储实验室和计算机位置，string用于存储该位置的用户姓名，若无人上机则为空字符串。

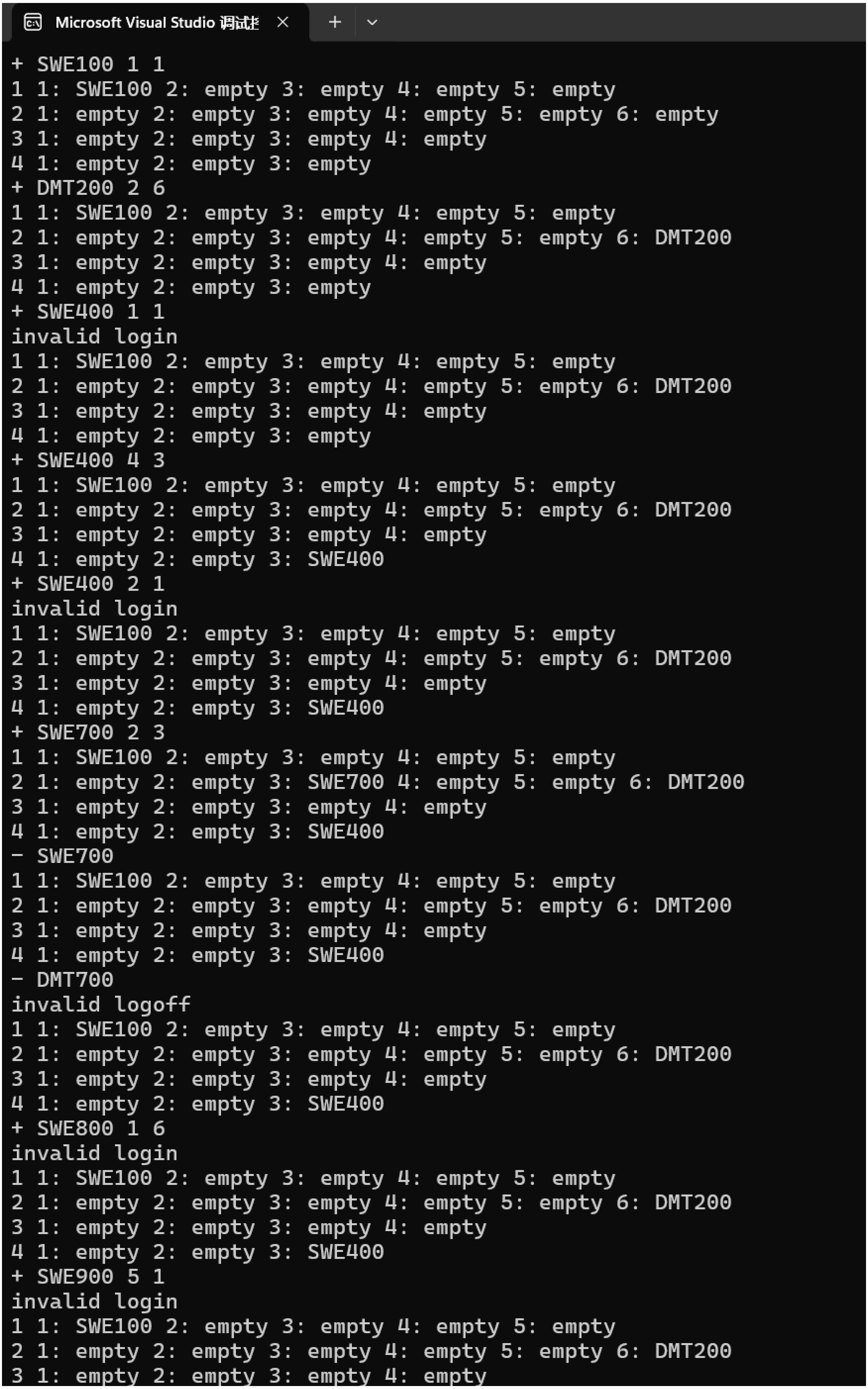
用户登入：先检查实验室编号与计算机编号是否合法，再遍历所有计算机确认该用户是否已经上机，最后确认登入的位置是否无人占用；如果上述都满足则将该位置string修改为用户名；如果不满足则输出invalid login。

用户登出：遍历所有计算机，检查该用户是否存在（上机）；如果找到则将该位置置为空字符串，找不到则输出invalid logoff。

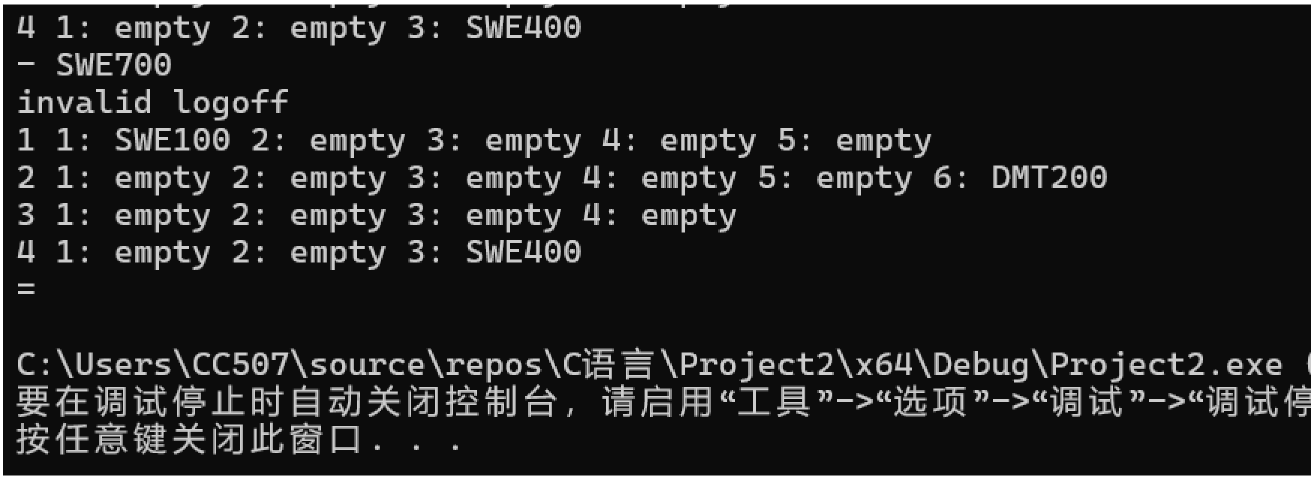
源码

```
#include<iostream>
#include<vector>
#include<unordered_map>
using namespace std;
vector<vector<string>> lab;
void print() {
    for (int i = 0; i < lab.size(); i++) {
        cout << i + 1 << " ";
        for (int j = 0; j < lab[i].size(); j++) {
            cout << j + 1 << ": ";
            if (lab[i][j] == "")cout << "empty";
            else cout << lab[i][j];
            cout << " ";
        }
        cout << endl;
    }
}
int main() {
    lab = {
        {"","","","",""},
        {"","","","","",""},
        {"","","",""},
        {"","",""}
    };
    string s;
    while (1) {
        cin >> s;
        if (s == "+") {
            string user;
            int a, b;
            cin >> user >> a >> b;
            a--; b--;
            int flag = 0;
            for (int i = 0; i < lab.size(); i++) {
                for (int j = 0; j < lab[i].size(); j++) {
                    if (lab[i][j] == user) {
                        flag = 1;
                    }
                }
            }
            if (a < lab.size() && b < lab[a].size() && lab[a][b] == "" && !flag) {
                lab[a][b] = user;
            }
            else {
                cout << "invalid login" << endl;
            }
        }
        else if (s == "-") {
            string user;
            cin >> user;
            int flag = 0;
            for (int i = 0; i < lab.size(); i++) {
                for (int j = 0; j < lab[i].size(); j++) {
                    if (lab[i][j] == user) {
                        lab[i][j] = "";
                        flag = 1;
                    }
                }
            }
            if (flag == 0) {
                cout << "invalid logoff" << endl;
            }
        }
        else if (s == "=") {
            break;
        }
        print();
    }
}
```

运行结果



```
+ SWE100 1 1
1 1: SWE100 2: empty 3: empty 4: empty 5: empty
2 1: empty 2: empty 3: empty 4: empty 5: empty 6: empty
3 1: empty 2: empty 3: empty 4: empty
4 1: empty 2: empty 3: empty
+ DMT200 2 6
1 1: SWE100 2: empty 3: empty 4: empty 5: empty
2 1: empty 2: empty 3: empty 4: empty 5: empty 6: DMT200
3 1: empty 2: empty 3: empty 4: empty
4 1: empty 2: empty 3: empty
+ SWE400 1 1
invalid login
1 1: SWE100 2: empty 3: empty 4: empty 5: empty
2 1: empty 2: empty 3: empty 4: empty 5: empty 6: DMT200
3 1: empty 2: empty 3: empty 4: empty
4 1: empty 2: empty 3: empty
+ SWE400 4 3
1 1: SWE100 2: empty 3: empty 4: empty 5: empty
2 1: empty 2: empty 3: empty 4: empty 5: empty 6: DMT200
3 1: empty 2: empty 3: empty 4: empty
4 1: empty 2: empty 3: SWE400
+ SWE400 2 1
invalid login
1 1: SWE100 2: empty 3: empty 4: empty 5: empty
2 1: empty 2: empty 3: empty 4: empty 5: empty 6: DMT200
3 1: empty 2: empty 3: empty 4: empty
4 1: empty 2: empty 3: SWE400
+ SWE700 2 3
1 1: SWE100 2: empty 3: empty 4: empty 5: empty
2 1: empty 2: empty 3: SWE700 4: empty 5: empty 6: DMT200
3 1: empty 2: empty 3: empty 4: empty
4 1: empty 2: empty 3: SWE400
- SWE700
1 1: SWE100 2: empty 3: empty 4: empty 5: empty
2 1: empty 2: empty 3: empty 4: empty 5: empty 6: DMT200
3 1: empty 2: empty 3: empty 4: empty
4 1: empty 2: empty 3: SWE400
- DMT700
invalid logoff
1 1: SWE100 2: empty 3: empty 4: empty 5: empty
2 1: empty 2: empty 3: empty 4: empty 5: empty 6: DMT200
3 1: empty 2: empty 3: empty 4: empty
4 1: empty 2: empty 3: SWE400
+ SWE800 1 6
invalid login
1 1: SWE100 2: empty 3: empty 4: empty 5: empty
2 1: empty 2: empty 3: empty 4: empty 5: empty 6: DMT200
3 1: empty 2: empty 3: empty 4: empty
4 1: empty 2: empty 3: SWE400
+ SWE900 5 1
invalid login
1 1: SWE100 2: empty 3: empty 4: empty 5: empty
2 1: empty 2: empty 3: empty 4: empty 5: empty 6: DMT200
3 1: empty 2: empty 3: empty 4: empty
```



```
4 1: empty 2: empty 3: SWE400
- SWE700
invalid logoff
1 1: SWE100 2: empty 3: empty 4: empty 5: empty
2 1: empty 2: empty 3: empty 4: empty 5: empty 6: DMT200
3 1: empty 2: empty 3: empty 4: empty
4 1: empty 2: empty 3: SWE400
=

C:\Users\CC507\source\repos\C语言\Project2\x64\Debug\Project2.exe (
要在调试停止时自动关闭控制台，请启用“工具”->“选项”->“调试”->“调试停
按任意键关闭此窗口. . .
```

题目4

实现思路

User类：

包含一个string类型变量name

无参构造函数默认将name置为empty便于后续输出

ComputerLab类：

包含一个vector<vector>类型变量computers，保存各个位置计算机的上机用户

is_user_exist(string name)方法：遍历所有位置，找到name的用户返回1，否则返回0

login(string name, int x, int y)方法：检查xy坐标是否合法，调用is_user_exist方法查找该用户是否已经上机，判断该位置用户是否为空；如果都合法则创建一个新的User(name)，并将该位置置为该User。

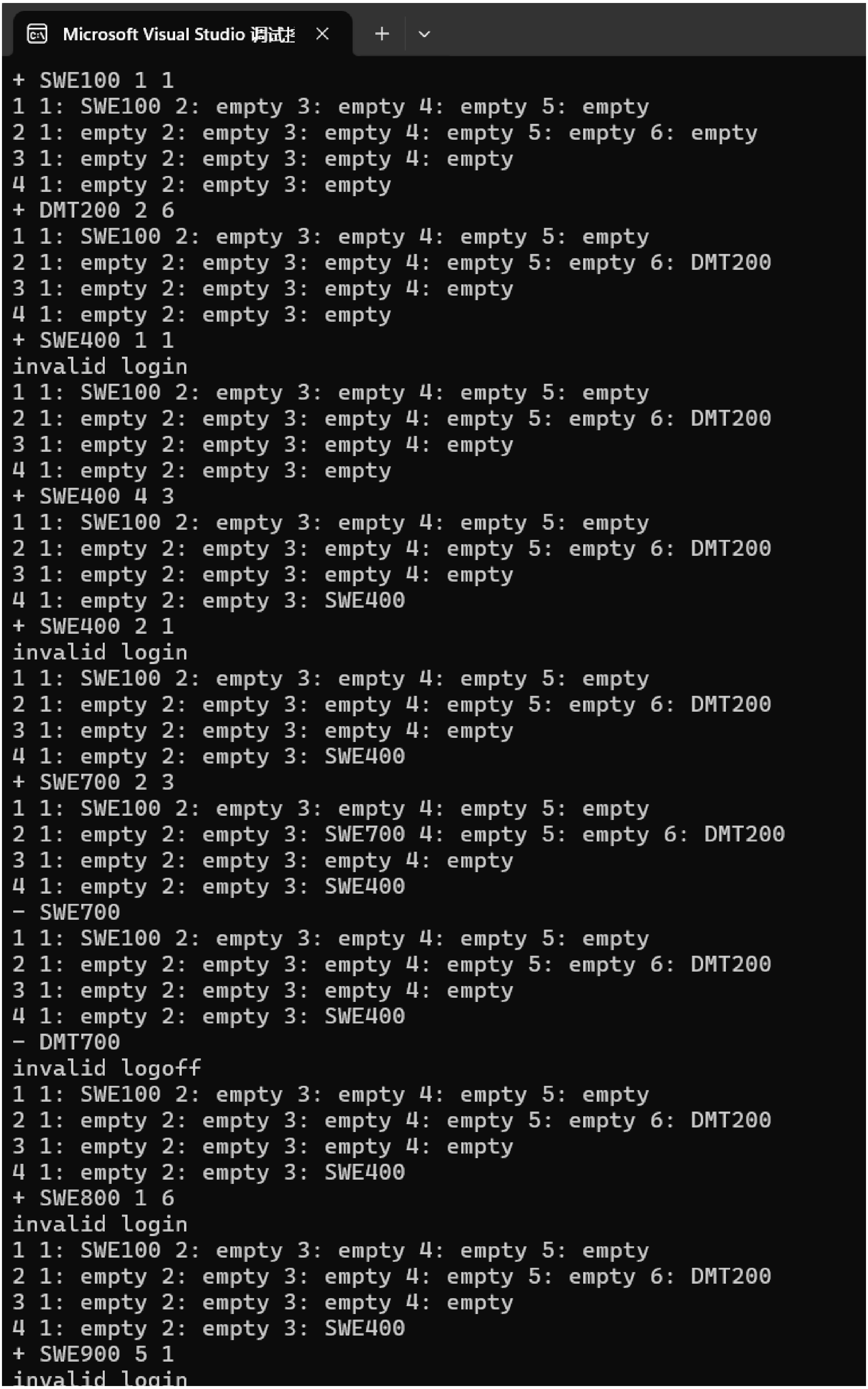
logout(string name)方法：与题目3类似。

print()方法，输出所有位置的上机用户。

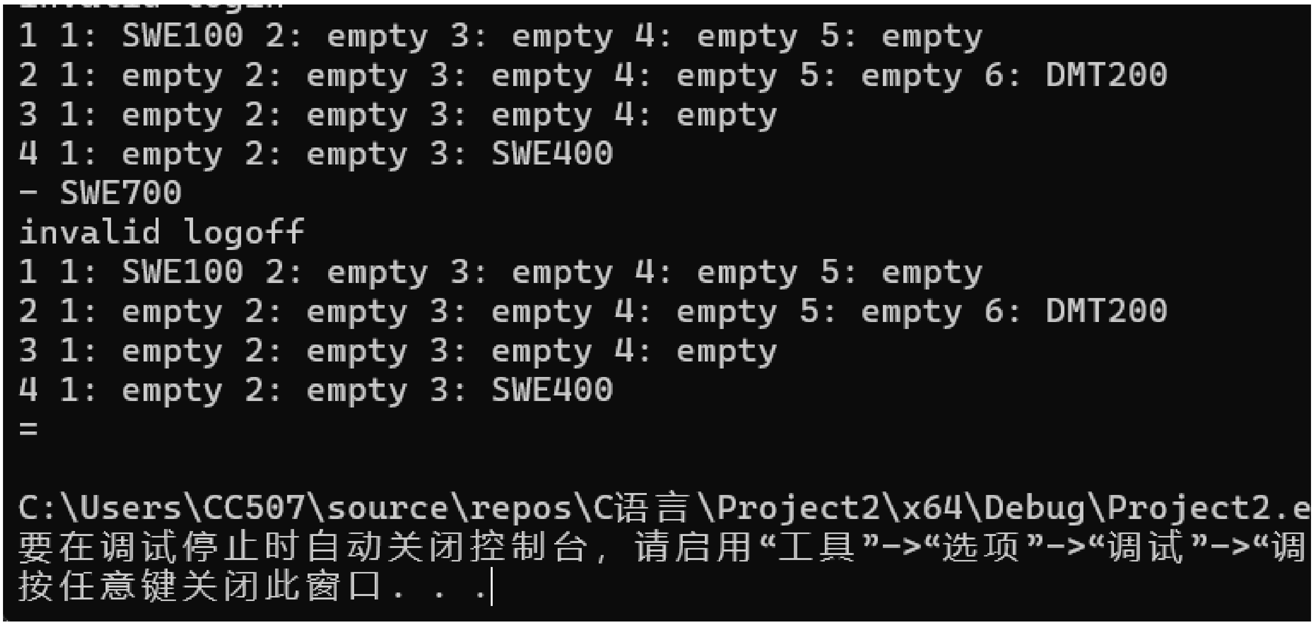
源码

```
#include<iostream>
#include<vector>
#include<string>
using namespace std;
class User
{
public:
    User() {
        this->name = "empty";
    }

    User(string name) {
        this->name = name;
    }

    string get_name() {
        return name;
    }

private:
    string name;
};
class ComputerLab
{
public:
    ComputerLab() {
        computers.resize(4);
        computers[0].resize(5);
        computers[1].resize(6);
        computers[2].resize(4);
        computers[3].resize(3);
    }

    bool is_user_exist(string name) {
        for (int i = 0; i < computers.size(); i++) {
            for (int j = 0; j < computers[i].size(); j++) {
                if (computers[i][j].get_name() == name) {
                    return 1;
                }
            }
        }
        return 0;
    }

    bool login(string name, int x, int y) {
        if (x > computers.size() || y > computers[x-1].size() || is_user_exist(name) || computers[x - 1][y - 1].get_name() != "empty") {
            cout << "invalid login" << endl;
            return 0;
        }
        User user(name);
        computers[x - 1][y - 1] = user;
        return 1;
    }

    bool logout(string name) {
        for (int i = 0; i < computers.size(); i++) {
            for (int j = 0; j < computers[i].size(); j++) {
                if (computers[i][j].get_name() == name) {
                    User emptyuser;
                    computers[i][j] = emptyuser;
                    return 1;
                }
            }
        }
        cout << "invalid logoff" << endl;
        return 0;
    }

    void print() {
    for (int i = 0; i < computers.size(); i++) {
        cout << i + 1 << " ";
        for (int j = 0; j < computers[i].size(); j++) {
            cout << j + 1 << ": " << computers[i][j].get_name() << " ";
        }
        cout << endl;
    }
}

private:
    vector<vector<User>> computers;
};
int main() {
    ComputerLab computerLab;
    while (1) {
        string s;
        cin >> s;
        if (s == "+") {
            string name;
            int x, y;
            cin >> name >> x >> y;
            computerLab.login(name, x, y);
        }
        else if (s == "-") {
            string name;
            cin >> name;
            computerLab.logout(name);
        }
        else if (s == "=") {
            break;
        }
        computerLab.print();
    }
}
```

运行结果



```
+ SWE100 1 1
1 1: SWE100 2: empty 3: empty 4: empty 5: empty
2 1: empty 2: empty 3: empty 4: empty 5: empty 6: empty
3 1: empty 2: empty 3: empty 4: empty
4 1: empty 2: empty 3: empty
+ DMT200 2 6
1 1: SWE100 2: empty 3: empty 4: empty 5: empty
2 1: empty 2: empty 3: empty 4: empty 5: empty 6: DMT200
3 1: empty 2: empty 3: empty 4: empty
4 1: empty 2: empty 3: empty
+ SWE400 1 1
invalid login
1 1: SWE100 2: empty 3: empty 4: empty 5: empty
2 1: empty 2: empty 3: empty 4: empty 5: empty 6: DMT200
3 1: empty 2: empty 3: empty 4: empty
4 1: empty 2: empty 3: empty
+ SWE400 4 3
1 1: SWE100 2: empty 3: empty 4: empty 5: empty
2 1: empty 2: empty 3: empty 4: empty 5: empty 6: DMT200
3 1: empty 2: empty 3: empty 4: empty
4 1: empty 2: empty 3: SWE400
+ SWE400 2 1
invalid login
1 1: SWE100 2: empty 3: empty 4: empty 5: empty
2 1: empty 2: empty 3: empty 4: empty 5: empty 6: DMT200
3 1: empty 2: empty 3: empty 4: empty
4 1: empty 2: empty 3: SWE400
+ SWE700 2 3
1 1: SWE100 2: empty 3: empty 4: empty 5: empty
2 1: empty 2: empty 3: SWE700 4: empty 5: empty 6: DMT200
3 1: empty 2: empty 3: empty 4: empty
4 1: empty 2: empty 3: SWE400
- SWE700
1 1: SWE100 2: empty 3: empty 4: empty 5: empty
2 1: empty 2: empty 3: empty 4: empty 5: empty 6: DMT200
3 1: empty 2: empty 3: empty 4: empty
4 1: empty 2: empty 3: SWE400
- DMT700
invalid logoff
1 1: SWE100 2: empty 3: empty 4: empty 5: empty
2 1: empty 2: empty 3: empty 4: empty 5: empty 6: DMT200
3 1: empty 2: empty 3: empty 4: empty
4 1: empty 2: empty 3: SWE400
+ SWE800 1 6
invalid login
1 1: SWE100 2: empty 3: empty 4: empty 5: empty
2 1: empty 2: empty 3: empty 4: empty 5: empty 6: DMT200
3 1: empty 2: empty 3: empty 4: empty
4 1: empty 2: empty 3: SWE400
+ SWE900 5 1
invalid login
```



```
invalid login
1 1: SWE100 2: empty 3: empty 4: empty 5: empty
2 1: empty 2: empty 3: empty 4: empty 5: empty 6: DMT200
3 1: empty 2: empty 3: empty 4: empty
4 1: empty 2: empty 3: SWE400
- SWE700
invalid logoff
1 1: SWE100 2: empty 3: empty 4: empty 5: empty
2 1: empty 2: empty 3: empty 4: empty 5: empty 6: DMT200
3 1: empty 2: empty 3: empty 4: empty
4 1: empty 2: empty 3: SWE400
=

C:\Users\CC507\source\repos\C语言\Project2\x64\Debug\Project2.e
要在调试停止时自动关闭控制台，请启用“工具”->“选项”->“调试”->“调
按任意键关闭此窗口. . .
```