

# 第一章

---

## 题目

---

年份

输入年份，判断是否为闰年。如果是，则输出yes，否则输出no

## 代码

---

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main(){
6      int year;
7
8      cout<<"请输入要判断的年份: "<<endl;
9      cin>>year;
10
11     if((year%4==0&&year%100!=0)|| (year%400==0)){
12         cout<<"yes"<<endl;
13     }else{
14         cout<<"no"<<endl;
15     }
16
17     return 0;
18 }
19
```

## 运行结果

---

```
请输入要判断的年份:
1950
no
```

```
请输入要判断的年份:
2000
yes
```

```
请输入要判断的年份:
2020
yes
```

# 第二章

---

## 题目

---

## 水仙花数

输出100~999中的所有水仙花数。若三位数ABC满足 $ABC=A^3+B^3+C^3$ ，则称其为水仙花数，例如 $153=1^3+5^3+3^3$ ，所以153就是水仙花数。

## 代码

```
1  #include <iostream>
2  #include <cmath>
3
4  using namespace std;
5
6  int main() {
7      int a, b, c;
8
9      for (int i = 100; i < 1000; i++) {
10         c = i % 10;
11         b = (i / 10) % 10;
12         a = (i / 100) % 10;
13
14         int tmp = (int)pow(a, 3) + pow(b, 3) + pow(c, 3);
15
16         if (tmp == i) {
17             cout << i << endl;
18         }
19         else {
20             continue;
21         }
22     }
23
24     return 0;
25 }
26
```

## 运行结果

```
153
370
371
407
```

## 第三章

### 题目

#### 分子量

给出一个物质的分子式（不带括号），求分子量。本题中的分子式只包含4中原子，分别为C,H,O,N,原子量为12.01,1.008,16.00, 14.01（单位：g/mol）。例如，C<sub>6</sub>H<sub>5</sub>OH的分子量为94.108g/mol。

## 代码

```
1  #include <iostream>
```

```
2  #include <string>
3
4  using namespace std;
5
6  int main()
7  {
8      double arr[] = {12.01, 1.008, 16.00, 14.01};
9      string s;
10
11     cout << "请输入化学式: " << endl;
12     cin >> s;
13
14     int len = s.size();
15     double total = 0.0;
16     int i = 0;
17
18     while(s[i]!='\0'){
19         int sum = 0;
20         char c = s[i];
21         i++;
22
23         while (s[i] >= '0' && s[i] <= '9') {
24             sum = sum * 10 + s[i] - '0';
25             i++;
26         }
27
28         if (sum != 0) {
29             if (c == 'C') {
30                 total += sum * arr[0];
31             }
32             else if (c == 'H') {
33                 total += sum * arr[1];
34             }
35             else if (c == 'O') {
36                 total += sum * arr[2];
37             }
38             else if (c == 'N') {
39                 total += sum * arr[3];
40             }
41         }
42         else{
43             if (c == 'C') {
44                 total += arr[0];
45             }
46             else if (c == 'H') {
47                 total += arr[1];
48             }
49             else if (c == 'O') {
50                 total += arr[2];
51             }
52             else if (c == 'N') {
53                 total += arr[3];
54             }
55         }
56     }
```

```

57
58     cout << "化学式对应的分子量为: " << endl;
59     cout << total << endl;
60
61     return 0;
62 }
63

```

## 运行结果

```

请输入化学式:
C6H12O6
化学式对应的分子量为:
180.156

```

```

请输入化学式:
H2O
化学式对应的分子量为:
18.016

```

```

请输入化学式:
C6H5OH
化学式对应的分子量为:
94.108

```

## 第四章

### 题目

#### 洪水

有一个 $n*m$  ( $1 \leq m, n < 30$ ) 的网格，每个格子是边长10米的正方形，网格四周是无限大的墙壁。输入每个格子的海拔高度，以及网格内雨水的总体积，输出水位的海拔高度以及有多少百分比的区域有水（即高度严格小于水平面）。

### 代码

```

1  #include <iostream>
2  #include <vector>
3  #include <iomanip>
4  #include <algorithm>
5
6  using namespace std;
7
8  int main() {
9      int m, n;
10     int sum[1000] = { 0 };
11     double water, rate, high;
12     double avg[1000] = { 0.0 };
13     vector<int> height;
14     height.push_back(0);
15

```

```

16     cout << "请输入m, n: " << endl;
17     cin >> m >> n;
18     cout << "请输入水量: " << endl;
19     cin >> water;
20     cout << "请输入每个格子的海拔高度: " << endl;
21     for (int i = 0; i < m * n; i++) {
22         int h;
23         cin >> h;
24         height.push_back(h);
25     }
26
27     sort(height.begin(), height.end());
28
29     sum[0] = 0;
30     int i = 1;
31     for (i = 1; i <= m * n; i++) {
32         sum[i] = sum[i - 1] + height[i];
33         avg[i] = (double)sum[i] / i;
34
35         if (((double)height[i+1] - avg[i]) * n * 100 > water) {
36             break;
37         }
38     }
39
40     rate = ((double)(i - 1)) / ((double)(m * n));
41     high = (double)(water / 100 / (i - 1)) + avg[i - 1];
42
43     cout << "水位的海拔高度为: " << high << endl;
44     cout << "所占区域的百分比为: " << setiosflags(ios::fixed) <<
45     setprecision(2) << rate * 100 << endl;
46
47     return 0;
48 }

```

## 运行结果

```

请输入m, n:
3 3
请输入水量:
10000
请输入每个格子的海拔高度:
25 37 45
51
12
34
94 83 27
水位的海拔高度为: 46.6667
所占区域的百分比为: 66.67

```

```
请输入m, n:
5 2
请输入水量:
10000
请输入每个格子的海拔高度:
15 98 102
56 44 39 75 23
28
31
水位的海拔高度为: 48
所占区域的百分比为: 70.00
```

```
请输入m, n:
2 3
请输入水量:
10000
请输入每个格子的海拔高度:
15 23 98 45 77 101
水位的海拔高度为: 69
所占区域的百分比为: 33.33
```

## 第五章

### 题目

#### 对称轴

给出平面上N ( $N \leq 1000$ ) 个点, 问是否可以找到一条竖线, 使得所有点左右对称。

### 代码

```
1  #include <iostream>
2  #include <set>
3
4  using namespace std;
5
6  int main() {
7      int n;
8      int sum = 0;
9      set<pair<int, int>> point;
10     bool flag = true;
11
12     cout << "请输入坐标点个数: " << endl;
13     cin >> n;
14
15     for (int i = 0; i < n; i++) {
16         int x, y;
17
18         cout << "请输入第" << i + 1 << "个点的x, y坐标: " << endl;
19         cin >> x >> y;
20     }
```

```

21         sum += x;
22
23         point.insert(pair<int, int>(n * x, y));
24     }
25
26     for (set<pair<int, int>>::iterator i = point.begin(); i != point.end();
27 i++) {
28         pair<int, int> p = *i;
29
30         if (point.find(pair<int, int> (2 * sum - p.first, p.second)) ==
31 point.end()) {
32             flag = false;
33             break;
34         }
35     }
36
37     if (flag) {
38         cout << "YES" << endl;
39     }
40     else {
41         cout << "NO" << endl;
42     }
43
44     return 0;
45 }
46

```

## 运行结果

```

请输入坐标点个数:
5
请输入第1个点的x, y坐标:
-2 5
请输入第2个点的x, y坐标:
6 5
请输入第3个点的x, y坐标:
0 0
请输入第4个点的x, y坐标:
2 3
请输入第5个点的x, y坐标:
4 0
YES

```

```
请输入坐标点个数：
4
请输入第1个点的x, y坐标：
0 4
请输入第2个点的x, y坐标：
0 0
请输入第3个点的x, y坐标：
4 0
请输入第4个点的x, y坐标：
2 3
NO
```

```
请输入坐标点个数：
8
请输入第1个点的x, y坐标：
1 9
请输入第2个点的x, y坐标：
1 1
请输入第3个点的x, y坐标：
2 -3
请输入第4个点的x, y坐标：
3 3
请输入第5个点的x, y坐标：
3 0
请输入第6个点的x, y坐标：
4 -3
请输入第7个点的x, y坐标：
5 1
请输入第8个点的x, y坐标：
5 9
YES
```

## 第六章

### 题目

#### 二叉树重建

输入一棵二叉树的先序遍历和中序遍历序列，输出后序遍历序列

### 代码

```
1  #include <iostream>
2  #include <string>
3
4  using namespace std;
5
6  string preorder, inorder;
7
8  void get_postorder(int s1,int s2,int length) {
9      if (length == 0) {
10         return;
```



```

11     }
12
13     int len = inorder.find(preorder[s1]) - s2;
14
15     get_postorder(s1 + 1, s2, len); //左子树
16     get_postorder(s1 + len + 1, s2 + len + 1, length - len - 1); //右子树
17
18     cout << preorder[s1];
19 }
20
21 int main(){
22     int len;
23
24     cout << "二叉树的先序遍历结果为: " << endl;
25     cin >> preorder;
26     cout << "二叉树的中序遍历结果为: " << endl;
27     cin >> inorder;
28
29     len = inorder.size();
30
31     cout << "二叉树的后序遍历结果为: " << endl;
32     get_postorder(0, 0, len);
33
34     return 0;
35 }
36

```

## 运行结果

```

二叉树的先序遍历结果为:
DBACEGF
二叉树的中序遍历结果为:
ABCDEFGF
二叉树的后序遍历结果为:
ACBFGED

```

```

二叉树的先序遍历结果为:
BCAD
二叉树的中序遍历结果为:
CBAD
二叉树的后序遍历结果为:
CDAB

```

```

二叉树的先序遍历结果为:
ABDCEF
二叉树的中序遍历结果为:
BDAECF
二叉树的后序遍历结果为:
DBEFCA

```