**实验三 问题二分析**

计算后一天日期的代码如下：

// 计算

QString MainWindow::calc(int year, int month, int day)

{

1 QString ans = "";

2 if (month != 2) { //非2月情况

3 if ( month == 4 || month == 6 || month == 9 || month == 11 ) { //小月只有30天

4 if (day == 30){

5 day = 1;

6 month++;

}

7 else day = day + 1;

}

8 else if(month != 12){ //非12月的大月，有31天

9 if (day == 31){

10 day = 1;

11 month++;

}

12 else day = day + 1;

}

13 else { //12月，涉及跨年

14 if (day == 31){

15 day = 1;

16 month = 1;

17 year++;

}

18 else day = day + 1;

}

}

19 else if (isLeapYear(year)) { //2月，分平年、闰年

20 if (day == 29){

21 day = 1;

22 month++;

}

23 else day = day + 1;

}

24 else { //平年2月只有28天

25 if (day == 28){

26 day = 1;

27 month++;

}

28 else day = day + 1;

}

29 QString newYear = QString::number(year);

30 QString newMonth = QString::number(month);

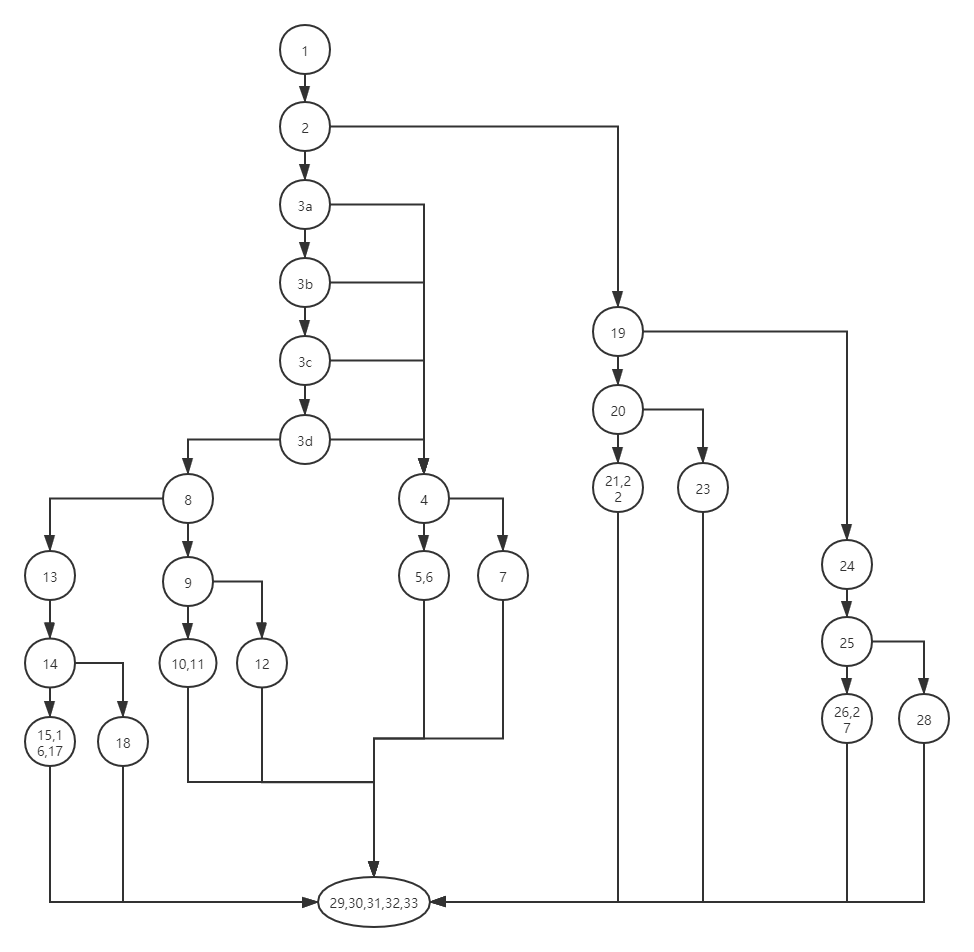
31 QString newDay = QString::number(day);

32 ans = newYear+"-"+newMonth+"-"+newDay;

33 return ans;

}

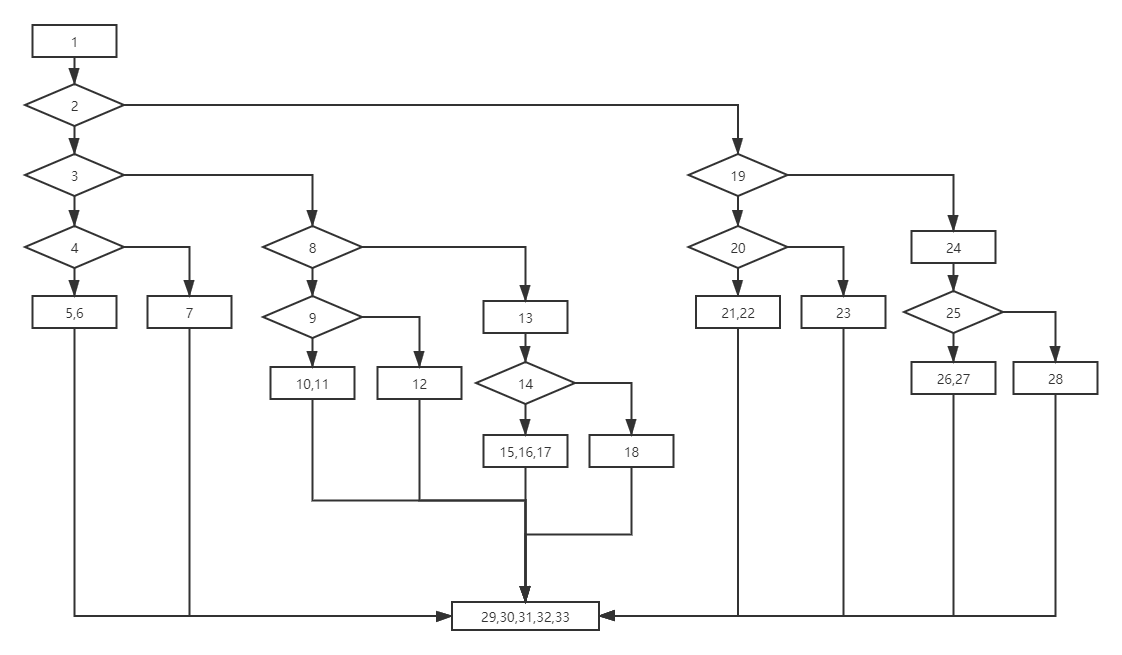
根据代码结构可以画出程序的控制流图如下：



1. **语句覆盖测试**

语句覆盖测试的要求是程序中所有的语句至少执行一次

程序流程图如下：

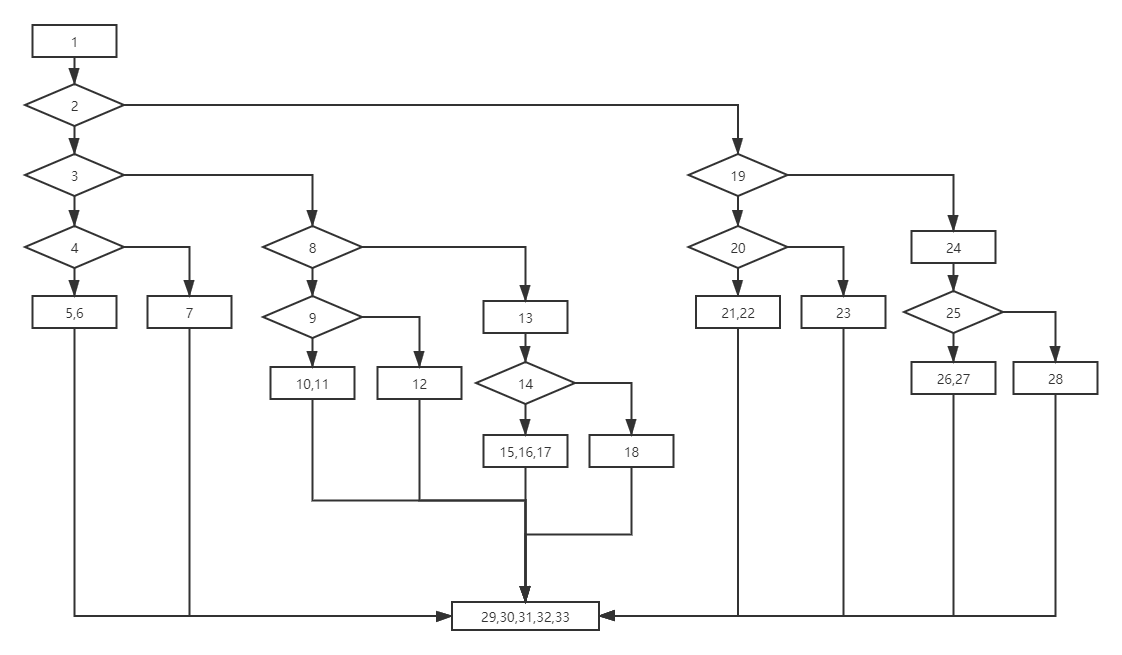


因此可以找出10条路径则可覆盖所有语句，因此可以设计测试用例

|  |  |  |
| --- | --- | --- |
| 测试用例 | 执行路径 | 预计输出 |
| 2020 4 30 | 1-2-3-4-5-6-29-30-31-32-33 | 2020 5 1 |
| 2020 4 15 | 1-2-3-4-7-29-30-31-32-33 | 2020 4 16 |
| 2020 1 31 | 1-2-3-8-9-10-11-29-30-31-32-33 | 2020 2 1 |
| 2020 1 15 | 1-2-3-8-9-12-29-30-31-32-33 | 2020 1 16 |
| 2020 12 31 | 1-2-3-8-13-14-15-16-17-29-30-31-32-33 | 2021 1 1 |
| 2020 12 15 | 1-2-3-8-13-14-18-29-30-31-32-33 | 2020 12 16 |
| 2020 2 29 | 1-2-19-20-21-22-29-30-31-32-33 | 2020 3 1 |
| 2020 2 15 | 1-2-19-20-23-29-30-31-32-33 | 2020 2 16 |
| 2019 2 28 | 1-2-19-24-25-26-27-29-30-31-32-33 | 2019 3 1 |
| 2019 2 15 | 1-2-19-24-25-28-29-30-31-32-33 | 2019 2 16 |

1. **判定覆盖测试**

判定覆盖测试要求所有判定分支都至少执行一次，即所有布尔表达式至少取真取假各一次

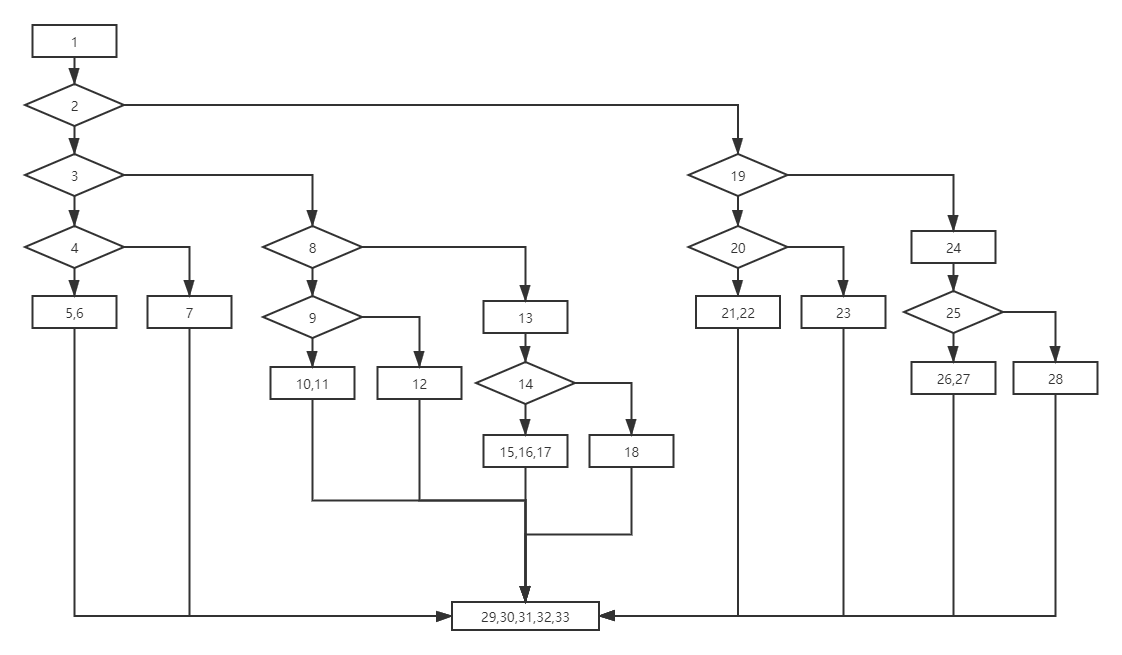


根据以上流程图可知，各个表达式取真取假各一次，则有10条执行路径

因此可以设计以下测试用例

|  |  |  |
| --- | --- | --- |
| 测试用例 | 执行路径 | 预计输出 |
| 2020 4 30 | 1-2-3-4-5-6-29-30-31-32-33 | 2020 5 1 |
| 2020 4 15 | 1-2-3-4-7-29-30-31-32-33 | 2020 4 16 |
| 2020 1 31 | 1-2-3-8-9-10-11-29-30-31-32-33 | 2020 2 1 |
| 2020 1 15 | 1-2-3-8-9-12-29-30-31-32-33 | 2020 1 16 |
| 2020 12 31 | 1-2-3-8-13-14-15-16-17-29-30-31-32-33 | 2021 1 1 |
| 2020 12 15 | 1-2-3-8-13-14-18-29-30-31-32-33 | 2020 12 16 |
| 2020 2 29 | 1-2-19-20-21-22-29-30-31-32-33 | 2020 3 1 |
| 2020 2 15 | 1-2-19-20-23-29-30-31-32-33 | 2020 2 16 |
| 2019 2 28 | 1-2-19-24-25-26-27-29-30-31-32-33 | 2019 3 1 |
| 2019 2 15 | 1-2-19-24-25-28-29-30-31-32-33 | 2019 2 16 |

1. **条件覆盖测试**



条件判断测试的要求是每个逻辑表达式至少取真取假各一次，因此可以按覆盖情况设计表：

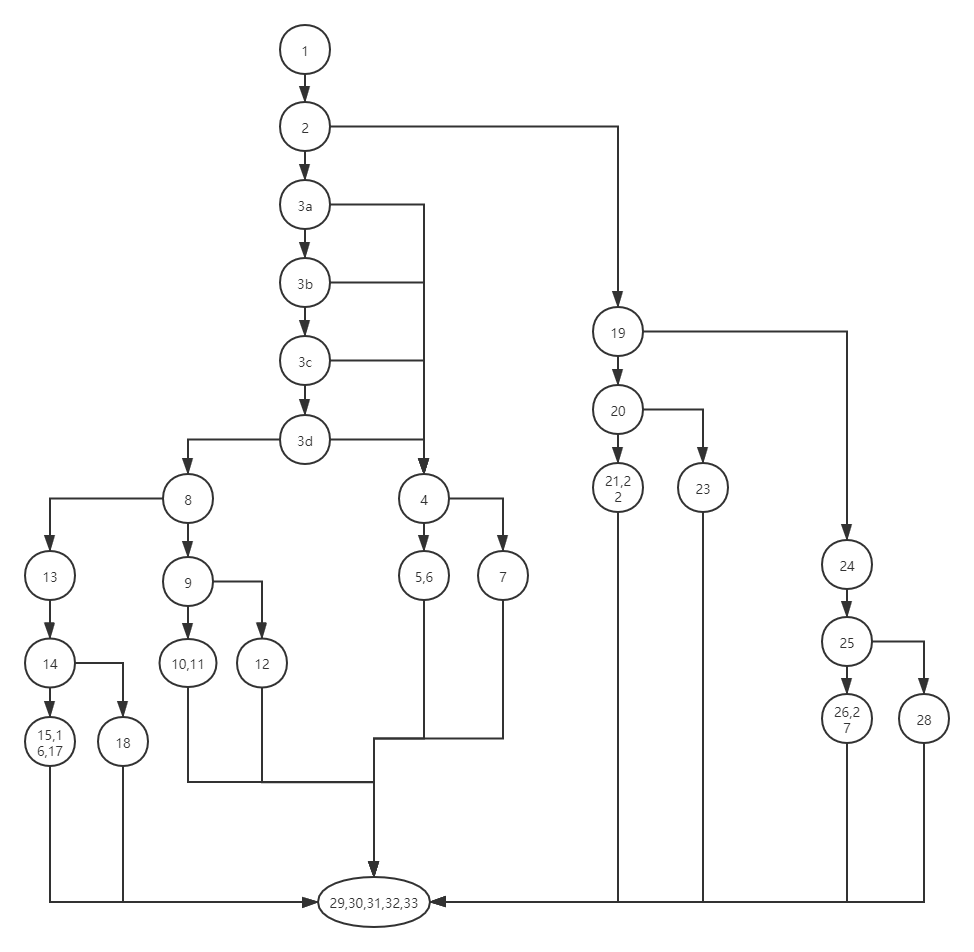
|  |  |  |
| --- | --- | --- |
|  | true | false |
| Month!=2 | 2020 1 31 | 2020 2 16 |
| Month==4 | 2020 4 16 | 2020 2 16 |
| Month==6 | 2020 6 16 | 2020 2 16 |
| Month==9 | 2020 9 16 | 2020 2 16 |
| Month==11 | 2020 11 16 | 2020 2 16 |
| Day==30 | 2020 4 30 | 2020 4 16 |
| Month!=12 | 2020 1 16 | 2020 12 16 |
| Day==31 | 2020 12 31 | 2020 12 16 |
| isLeapYear(year) | 2020 2 16 | 2019 2 16 |
| Day==29 | 2020 2 29 | 2020 2 16 |
| Day==28 | 2019 2 28 | 2019 2 16 |

由上表可得出测试用例

|  |  |  |
| --- | --- | --- |
| 测试用例 | 执行路径 | 预计输出 |
| 2020 4 16 | 1-2-3-4-7-29-30-31-32-33 | 2020 4 17 |
| 2020 6 16 | 1-2-3-4-7-29-30-31-32-33 | 2020 7 17 |
| 2020 9 16 | 1-2-3-4-7-29-30-31-32-33 | 2020 9 17 |
| 2020 11 16 | 1-2-3-4-7-29-30-31-32-33 | 2020 11 17 |
| 2020 4 30 | 1-2-3-4-5-6-29-30-31-32-33 | 2020 5 1 |
| 2020 1 16 | 1-2-3-8-9-12-29-30-31-32-33 | 2020 1 17 |
| 2020 1 31 | 1-2-3-8-9-10-11-29-30-31-32-33 | 2020 2 1 |
| 2020 12 16 | 1-2-3-8-13-14-18-29-30-31-32-33 | 2020 12 17 |
| 2020 12 31 | 1-2-3-8-13-14-15-16-17-29-30-31-32-33 | 2021 1 1 |
| 2020 2 16 | 1-2-19-20-23-29-30-31-32-33 | 2020 2 17 |
| 2019 2 16 | 1-2-19-24-25-28-29-30-31-32-33 | 2019 2 17 |
| 2020 2 29 | 1-2-19-20-21-22-29-30-31-32-33 | 2020 3 1 |
| 2019 2 28 | 1-2-19-24-25-26-27-29-30-31-32-33 | 2019 3 1 |

1. **基本路径测试法**

程序控制流图如下：



根据程序流图，可以的出环复杂度：

V(G)=封闭区间数=判定结点数+1=13

根据基本路径集，筛去不可能路径，可以得到以下独立路径：

1-2-3-4-5-6-29-30-31-32-33

1-2-3-4-7-29-30-31-32-33

1-2-3-8-9-10-11-29-30-31-32-33

1-2-3-8-9-12-29-30-31-32-33

1-2-3-8-13-14-15-16-17-29-30-31-32-33

1-2-3-8-13-14-18-29-30-31-32-33

1-2-19-20-21-22-29-30-31-32-33

1-2-19-20-23-29-30-31-32-33

1-2-19-24-25-26-27-29-30-31-32-33

1-2-19-24-25-28-29-30-31-32-33

因此可以设计测试用例如下：

|  |  |  |
| --- | --- | --- |
| 测试用例 | 执行路径 | 预计输出 |
| 2020 4 30 | 1-2-3-4-5-6-29-30-31-32-33 | 2020 5 1 |
| 2020 4 15 | 1-2-3-4-7-29-30-31-32-33 | 2020 4 16 |
| 2020 1 31 | 1-2-3-8-9-10-11-29-30-31-32-33 | 2020 2 1 |
| 2020 1 15 | 1-2-3-8-9-12-29-30-31-32-33 | 2020 1 16 |
| 2020 12 31 | 1-2-3-8-13-14-15-16-17-29-30-31-32-33 | 2021 1 1 |
| 2020 12 15 | 1-2-3-8-13-14-18-29-30-31-32-33 | 2020 12 16 |
| 2020 2 29 | 1-2-19-20-21-22-29-30-31-32-33 | 2020 3 1 |
| 2020 2 15 | 1-2-19-20-23-29-30-31-32-33 | 2020 2 16 |
| 2019 2 28 | 1-2-19-24-25-26-27-29-30-31-32-33 | 2019 3 1 |
| 2019 2 15 | 1-2-19-24-25-28-29-30-31-32-33 | 2019 2 16 |