1 scanf(“%d %d %d”,&x, &y, &z);

2 if (y < 0 and x > 0)

3 pow = -y;

4 else pow = y;

5 z = pow - x;

6 if (pow != 0) {

7 z = z \* x;

8 pow = pow - 1;}

9 if (z < x \* y and z > 0)

10 z = 1.0 / z;

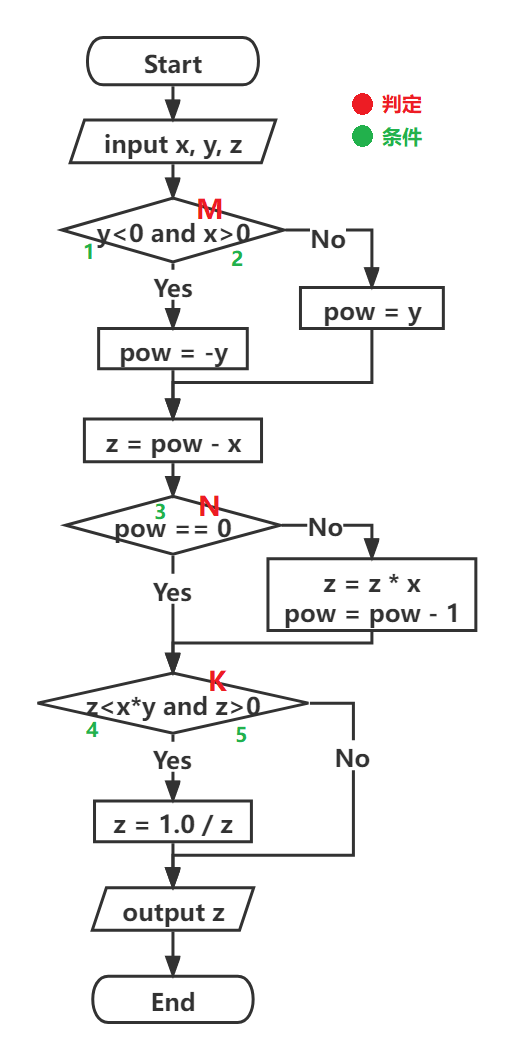
11 printf (“%f”, z);

测试循环时，认为它只执行一次。

1. 用判定条件覆盖法完成测试用例设计。
2. 用基本路径法完成测试用例设计。
3. 对变量z进行数据流测试，找出z的定义节点、使用节点、定义清除路径，并分析其存在的问题。

**1、用判定条件覆盖法完成测试用例设计。**

画出程序流程图：



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **输入** | | | **取值条件** | **判定条件** | **预测输出** |
| **x** | **y** | **z** |
| 1 | 5 | -3 | 0 | T1, T2, F3, F4, F5 | M=T, N=F, K=F | -10.000000 |
| 2 | 3 | 10 | 0 | F1, T2, F3, T4,T5 | M=F, N=F, K=T | 0.047619 |
| 3 | -9 | 0 | 0 | F1, F2, T3, F4,T5 | M=F, N=T, K=F | 9.000000 |

所有条件的可能取值已至少执行一次，所有判定的可能结果已至少执行一次。

附验证代码：

#include <stdio.h>

int main() {

    double x, y, z, pow;

    scanf("%lf %lf %lf", &x, &y, &z);

    if( y < 0 ) printf("T1\n");

    else printf("F1\n");

    if( x > 0 ) printf("T2\n");

    else printf("F2\n");

    if( y<0 && x>0 ) {

        pow = -y;

        printf("M=T\n");

    } else {

        pow = y;

        printf("M=F\n");

    }

    z = pow - x;

    if( pow == 0 ) printf("T3\n");

    else printf("F3\n");

    if( pow != 0 ) {

        z = z \* x;

        pow = pow - 1;

        printf("N=F\n");

    } else {

        printf("N=T\n");

    }

    if( z < x\*y ) printf("T4\n");

    else printf("F4\n");

    if( z > 0 ) printf("T5\n");

    else printf("F5\n");

    if( z<x\*y && z>0 ) {

        z = 1.0 / z;

        printf("K=T\n");

    } else {

        printf("K=F\n");

    }

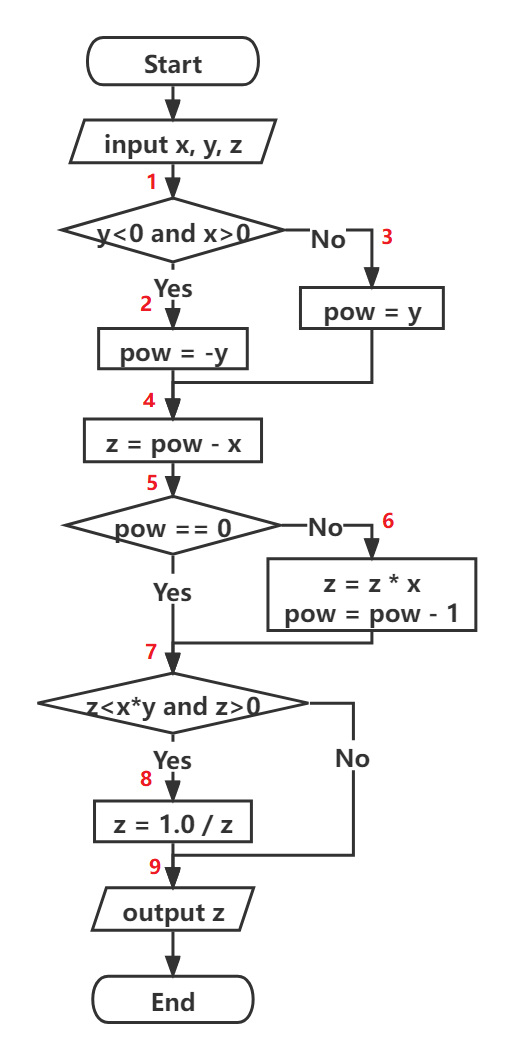
    printf("%f", z);

    return 0;

}

**2、用基本路径法完成测试用例设计。**

程序流程图：



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **输入** | | | **覆盖路径** | **覆盖条件** | **预测输出** |
| **x** | **y** | **z** |
| 1 | 5 | -3 | 0 | 1-2-4-5-6-7-9 | T1, T2, F3, F4, F5 | -10.000000 |
| 2 | 3 | 10 | 0 | 1-3-4-5-6-7-8-9 | F1, T2, F3, T4,T5 | 0.047619 |
| 3 | -9 | 0 | 0 | 1-3-4-5-7-9 | F1, F2, T3, F4,T5 | 9.000000 |

可以看到，所有路径均已执行。

附验证代码：

#include <stdio.h>

int main() {

    double x, y, z, pow;

    scanf("%lf %lf %lf", &x, &y, &z);

    printf("path:1\n");

    if( y < 0 ) printf("T1\n");

    else printf("F1\n");

    if( x > 0 ) printf("T2\n");

    else printf("F2\n");

    if( y<0 && x>0 ) {

        pow = -y;

        printf("path:2\n");

    } else {

        pow = y;

        printf("path:3\n");

    }

    z = pow - x;

    printf("path:4\n");

    printf("path:5\n");

    if( pow == 0 ) printf("T3\n");

    else printf("F3\n");

    if( pow != 0 ) {

        z = z \* x;

        pow = pow - 1;

        printf("path:6\n");

    }

    printf("path:7\n");

    if( z < x\*y ) printf("T4\n");

    else printf("F4\n");

    if( z > 0 ) printf("T5\n");

    else printf("F5\n");

    if( z<x\*y && z>0 ) {

        z = 1.0 / z;

        printf("path:8\n");

    }

    printf("path:9\n");

    printf("%f", z);

    return 0;

}

**3、对变量z进行数据流测试，找出z的定义节点、使用节点、定义清除路径，并分析其存在的问题。**

1 scanf(“%d %d %d”,&x, &y, &z);

2 if (y < 0 and x > 0)

3 pow = -y;

4 else pow = y;

5 z = pow - x;

6 if (pow != 0) {

7 z = z \* x;

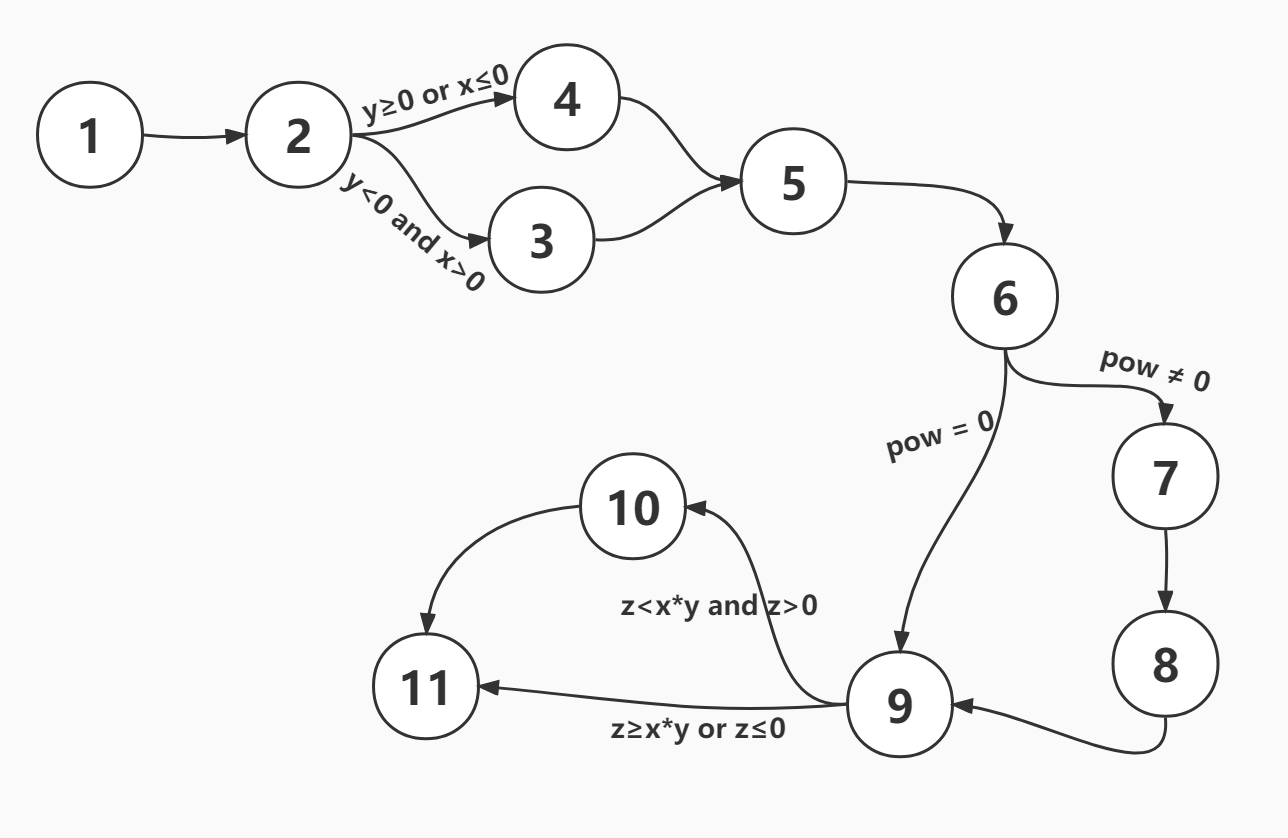
8 pow = pow - 1;}

9 if (z < x \* y and z > 0)

10 z = 1.0 / z;

11 printf (“%f”, z);

画出控制流图：



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **变量** | **定义节点** | **使用节点** | | **定义使用对** | **定义使用路径（是否为定义-清除路径）** |
| **计算使用** | **谓词使用** |
| z | 1, 5, 7, 10 | 7, 10 | 9, 11 | (1, 7)  (1, 9)  (1, 10)  (1, 11) | 1-2-3-5-6-7（F）  1-2-4-5-6-7（F）  1-2-3-5-6-9（F）  1-2-4-5-6-9（F）  1-2-3-5-6-7-8-9（F）  1-2-4-5-6-7-8-9（F）  1-2-3-5-6-7-8-9-10（F）  1-2-4-5-6-7-8-9-10（F）  1-2-3-5-6-9-10（F）  1-2-4-5-6-9-10（F）  1-2-3-5-6-9-11（F）  1-2-4-5-6-9-11（F）  1-2-3-5-6-7-8-9-10-11（F）  1-2-4-5-6-7-8-9-10-11（F）  1-2-3-5-6-7-8-9-11（F）  1-2-4-5-6-7-8-9-11（F） |
| (5, 7) | **5-6-7（T）** |
| (5, 9) | 5-6-7-8-9（F）  **5-6-9（T）** |
| (5, 10) | 5-6-7-8-9-10（F）  **5-6-9-10（T）** |
| (5, 11) | 5-6-9-10-11（F）  **5-6-9-11（T）**  5-6-7-8-9-10-11（F）  5-6-7-8-9-11（F） |
| (7, 9) | **7-8-9（T）** |
| (7, 10) | **7-8-9-10（T）** |
| (7, 11) | 7-8-9-10-11（F）  **7-8-9-11（T）** |
| (10, 11) | **10-11（T）** |

存在问题：

（1）语句1输入数据的数据类型错误，应该为浮点类型。

（2）语句1和语句5都是z的定义节点，但在语句1和语句5之间没有z的使用节点，相当于语句1做了无效定义。