Министерство науки и высшего образования Российской Федерации
Калужский филиал
федерального государственного бюджетного
образовательного учреждения высшего образования
«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»
(КФ МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИУК «Информатика и управление»

КАФЕДРА ИУК4 «Программное обеспечение ЭВМ, информационные технологии»

ЛАБОРАТОРНАЯ РАБОТА №2

«ТЕСТИРОВАНИЕ ПРОГРАММ МЕТОДАМИ «БЕЛОГО ЯЩИКА»»

ДИСЦИПЛИНА: «Тестирование и отладка программного обеспечения»

Выполнил: студент гр. ИУК4-82Б _____ (подпись) (Панина В.С. (Ф.И.О.))

Проверил: _____ (подпись) (Красавин Е.В. (Ф.И.О.))

Дата сдачи (защиты):

Результаты сдачи (защиты):

- Балльная оценка:
- Оценка:

Калуга, 2021

Цель: формирование практических навыков составления набора тестовых данных для структурного тестирования.

Задачи:
для заданной процедуры составить граф–схему и тестовые наборы для тестирования маршрутов по определенным критериям.

Задача 1. Определить цикломатическую сложность потоковых графов, представленных на рис. 1.

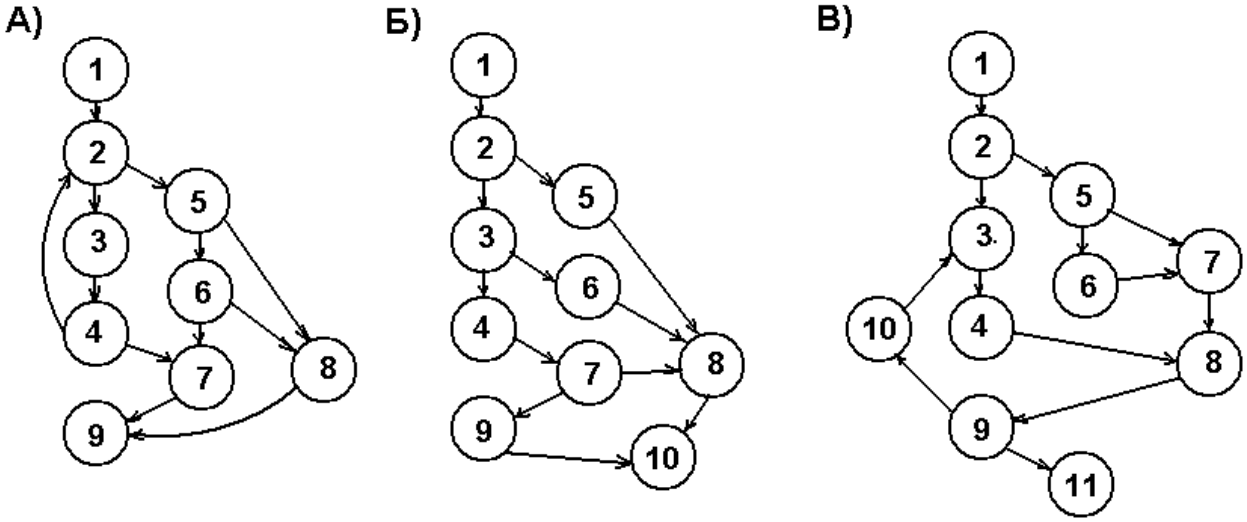

Рис. 1. Потоковые графы

Цикломатическая сложность определяется по формуле:

$$V(G) = E - N + 2$$

где E − количество дуг, N − количество узлов потокового графа;

А: $E = 12, \quad N = 9 \Rightarrow V(G) = 12 - 9 + 2 = 5;$

Б: $E = 12, \quad N = 10 \Rightarrow V(G) = 12 - 10 + 2 = 4;$

В: $E = 13, \quad N = 11 \Rightarrow V(G) = 13 - 11 + 2 = 4;$

Задача 2. В соответствии с концепцией максимально полного тестирования всех маршрутов программы определить независимые пути потоковых графов, представленных на рис.1.

А) Независимые пути:

Путь 1: 1-2-3-4-7-9
Путь 2: 1-2-5-6-7-9
Путь 3: 1-2-5-6-8-9
Путь 4: 1-2-5-8-9
Путь 5: 1-2-3-4-2-3-4-7-9

Б) Независимые пути:

Путь 1: 1-2-3-4-7-9-10
Путь 2: 1-2-3-4-7-8-10
Путь 3: 1-2-5-8-10

Путь 4: 1-2-3-6-8-10

В) Независимые пути:

Путь 1: 1-2-3-4-8-9-11
Путь 2: 1-2-5-6-7-8-9-11
Путь 3: 1-2-5-7-8-9-11
Путь 4: 1-2-5-7-8-9-10-3-4-8-9-11

Задача 3. Для заданной процедуры (варианты 1-6) составить потоковый граф, определить цикломатическую сложность потокового графа по каждой из трех формул и составить тестовые наборы по критерию покрытия маршрутов.

Вариант 5.

```
procedure m(a,b: real; var x: real)
begin
if (a>0)and(b<0) then x:=x+1
else if ((a=2)or(x>3))and(b>-10) then x:=x-1;
end;
```

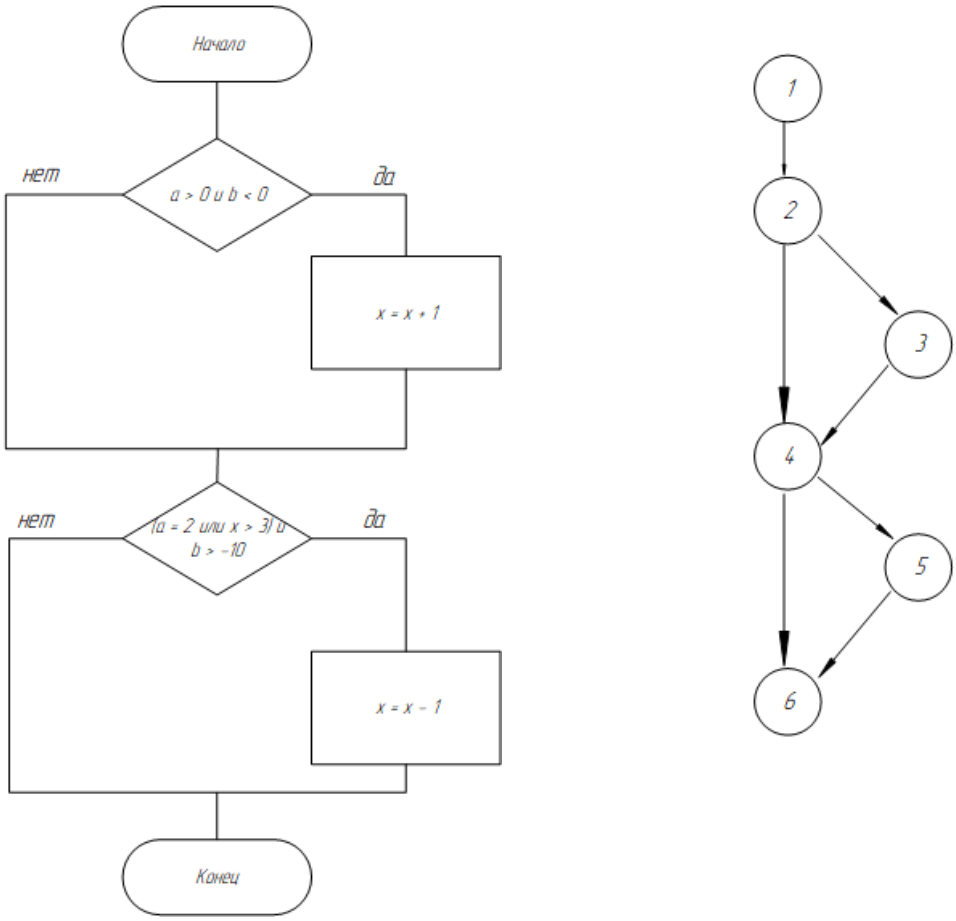

Рис.2. Блок-схема и потоковый граф алгоритма

Цикломатическая сложность алгоритма:

1) $V(G) = 3$ региона;
2) $V(G) = 7$ дуг - 6 узлов $+ 2 = 3$;
3) $V(G) = 2$ предикатных узлов $+ 1 = 3$.

Тестовые наборы:

Тест №1

Исходные данные (ИД): $a = 0, b = -1, x = 2$

Ожидаемые результаты (ОЖ.РЕЗ.): $a = 0, b = -1, x = 2$

Тест №2

Исходные данные (ИД): $a = 1, b = -1, x = 3$

Ожидаемые результаты (ОЖ.РЕЗ.): $a = 1, b = -1, x = 4$

Тест №3

Исходные данные (ИД): $a = 2, b = 1, x = 4$

Ожидаемые результаты (ОЖ.РЕЗ.): $a = 2, b = 1, x = 3$

Задача 4. Для заданной процедуры (варианты 1-6) составить граф-схему и тестовые наборы для тестирования маршрутов по критериям:

1. покрытия операторов;
2. покрытия решений (переходов);
3. покрытия условий;
4. покрытия решений/условий;
5. комбинаторного покрытия условий.

Проанализируйте целесообразность каждого из критериев для своей программы, укажите их недостатки, достоинства и преимущества над другими критериями.

По приведенным листингам программного кода напишите собственную программу и проверьте тестовые наборы.

Скриншоту выполнения программ приведите в отчете.

Решение:

1. Критерий покрытия операторов подразумевает такой подбор тестов, чтобы каждый оператор программы выполнялся, по крайней мере, один раз.

Покрытие операторов будет реализовано при любом

$$a > 0, \quad -10 < b < 0, \quad x \geq 3$$

(путь 1-2-3-4-5-6).

2. Критерий покрытия решений (переходов) подразумевает такое количество и состав тестов, чтобы результат проверки каждого условия (т.е. решение) принимал значения «истина» или «ложь», по крайней мере, один раз. Таким образом, рассматриваемую программу можно протестировать двумя тестами, покрывающими либо пути: 1-2-3-4-5-6, 1-2-3-4.

3. Покрытие условий формируют некоторое количество тестов, достаточное для того, чтобы все возможные результаты каждого условия в решении были выполнены, по крайней мере, один раз.

Покрытие условий проверяет пять условий:

l) $a > 0$;

2) $b < 0$;

3) $a = 2$;

4) $x > 3$;

5) $b > -10$.

Необходимо реализовать все возможные ситуации:

Тесты, удовлетворяющие этому условию:

$a = 2, b = -1, x = 3$ – путь 1-2-3-4-5-6, условия: 1-да, 2-да, 3-да, 4-да, 5-да

$a = 0, b = 1, x = 1$ – путь 1-2-3-4, условия: 1-нет, 2-нет, 3–нет, 4-нет, 5-да

Основной недостаток метода – недостаточная чувствительность к ошибкам в логических выражениях.

4. Покрытие решений/условий. Тесты должны составляться так, чтобы, по крайней мере, один раз выполнились все возможные результаты каждого условия и все результаты каждого решения, и каждому оператору управление передавалось, по крайней мере, один раз.

Тесты, удовлетворяющие этому условию:

$a = 2, b = -1, x = 3$ – путь 1-2-3-4-5-6, условия: 1-да, 2-да, 3-да, 4-да, 5-да

$a = 0, b = 1, x = 1$ – путь 1-2-3-4, условия: 1-нет, 2-нет, 3–нет, 4-да

Достоинство: гарантирует более полное тестовое покрытие по сравнению с предыдущим методом.

5. Комбинаторное покрытие условий требует покрыть тестами все возможные комбинации результаты условий:

1) $a > 0, b < 0$

2) $a > 0, b \geq 0$

3) $a \leq 0, b < 0$

4) $a \leq 0, b \geq 0$

5) $a = 2, x \geq 3, b > -10$

6) $a = 2, x > 3, b \leq -10$

7) $a = 2, x \leq 3, b > -10$

8) $a = 2, x \leq 3, b \leq -10$

9) $a \neq 2, x > 3, b > -10$

10) $a \neq 2, x > 3, b \leq -10$

11) $a \neq 2, x \leq 3, b > -10$

12) $a \neq 2, x \leq 3, b \leq -10$

Эти комбинации можно проверить восемью тестами:

$a = 2, b = -9, x = 4$ — проверяет комбинации (1), (5);

$a = 2, b = -11, x = 4$ — проверяет комбинации (1), (6);

$a = 2, b = 0, x = 3$ — проверяет комбинации (2), (7);

$a = 2, b = -11, x = 3$ — проверяет комбинации (1), (8);

$a = 0, b = -9, x = 4$ — проверяет комбинации (3), (9);

$a = 0, b = -11, x = 4$ — проверяет комбинации (3), (10);
$a = 0, b = 0, x = 3$ — проверяет комбинации (4), (11);
$a = 0, b = -11, x = 3$ — проверяет комбинации (3), (12).

Листинг программы:

```
def func(a, b, x):
    if (a > 0) and (b < 0):
        x += 1
    if (a == 2 or x > 3) and (b > -10):
        x -= 1
    return x
```

Результаты работы программы:

```
a =  2  b =  -9  x =  4 Результат x =  4
a =  2  b =  -11  x =  4 Результат x =  5
a =  2  b =  0  x =  3 Результат x =  2
a =  2  b =  -11  x =  3 Результат x =  4
a =  0  b =  -9  x =  4 Результат x =  3
a =  0  b =  -11  x =  4 Результат x =  4
a =  0  b =  0  x =  3 Результат x =  3
a =  0  b =  -11  x =  3 Результат x =  3
```

Рис. 3. Результат работы

Задача 5. Определить типы циклов в потоковых графах, представленных на рис.1, 4.

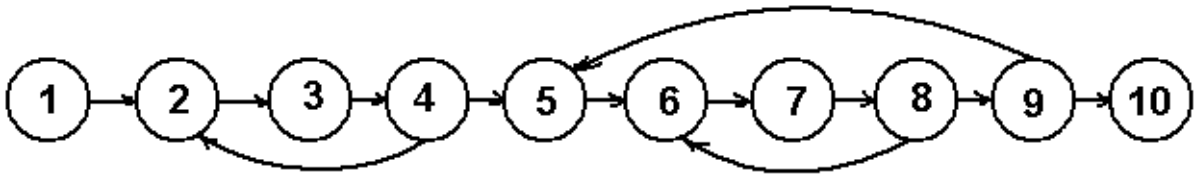

Рис. 4. Потоковый граф

Рис.1.
А) 2-3-4-2 – простой цикл
Б) циклов нет
В) 3-4-8-9-10-3 – простой цикл

Рис.4.
Цикл 2-3-4-2 – простой цикл
Цикл 6-7-8-6 – вложенный цикл
Цикл 5-6-7-8-9 – объемлющий цикл

Задача 6. Сколько наборов тестов необходимо для тестирования программы, потоковый граф которой представлен на рис. 4.

Для участка 5-9 используем методику тестирования вложенных циклов. Простой цикл будем тестировать на минимальных и максимальных значениях параметра цикла, устанавливая минимальным параметр внешнего цикла, затем перейдем к внешнему. Получится 3 тестовых набора.

С учетом простого цикла 2-3-4-2 всего потребуется 5 наборов тестов.

Выводы: в результате выполнения лабораторной работы были сформированы практические навыки составления набора тестовых данных для структурного тестирования.