## Part 1

6. Answer questions a–c for the graph in Figure 7.2.

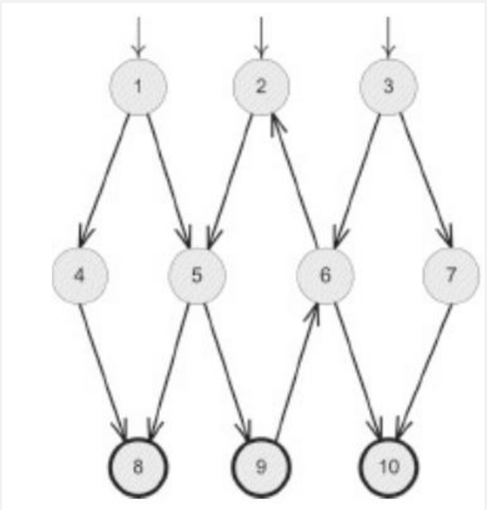


**Figure 7.2.** Example of paths.

a) List the test requirements for Node Coverage, Edge Coverage, and Prime Path Coverage on the graph.

Node coverage: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10

Edge coverage: (1,4), (1,5), (4,8), (5,8), (5,9), (2,5), (9,6), (6,2), (3,6), (3,7), (6,10), (7,10)

Prime path coverage: (1,4,8), (1,5,8), (3,6,10), (3,7,10), (1,5,9,6,2), (1,5,9,6,10), (2,5,9,6,10), (2,5,9,6,2), (3,6,2,5,8), (3,6,2,5,9), (5,9,6,2,5), (6,2,5,9,6), (9,6,2,5,8), (9,6,2,5,9)

b) List test paths that achieve Node Coverage but not Edge Coverage on the graph.

(1,4,8) (1,5,9) (2,6,10) (3,7,10)

c) List test paths that achieve Edge Coverage but not Prime Path Coverage on the graph.

(1,4,8) (1,5,9,6,2,5,8) (3,6,10) (3,7,10)

7. Answer questions a-d for the graph defined by the following sets:

- N = {1, 2, 3}
- N0 = {1}
- Nf = {3}
- E = {(1, 2), (1, 3), (2, 1), (2, 3), (3, 1)}

Also consider the following (candidate) paths:

- p1 = [1, 2, 3, 1]
- p2 = [1, 3, 1, 2, 3]
- p3 = [1, 2, 3, 1, 2, 1, 3]
- p4 = [2, 3, 1, 3]
- p5 = [1, 2, 3, 2, 3]

(a) Which of the listed paths are test paths? For any path that is not a test path, explain why not.

p2, p3, p5 test paths must start with a node in N0 and end with a node in Nf

(b) List the eight test requirements for Edge-Pair Coverage (only the length two subpaths).

| | | | |
|---|---|---|---|
| (1,2,1) | (1,2,3) | (1,3,1) | (2,1,3) |
| (2,1,2) | (2,3,1) | (3,1,2) | (3,1,3) |

(c) Does the set of test paths from part (a) above satisfy Edge-Pair Coverage? If not, state what is missing.

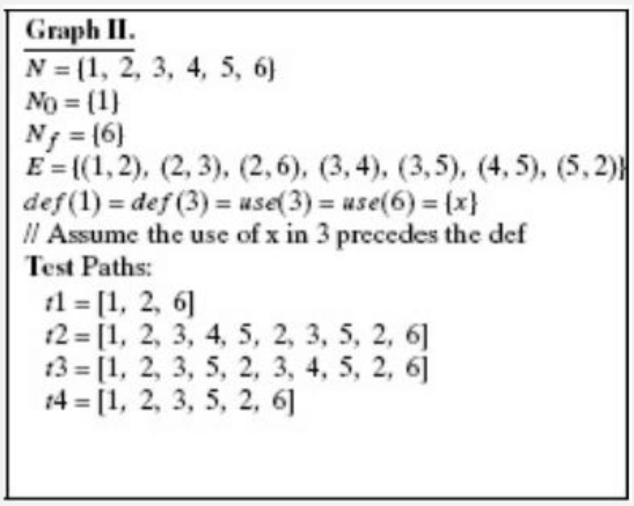
No, (2,1,2) and (3,1,3) are missing

(d) Consider the prime path [3, 1, 3] and path p2. Does p2 tour the prime path directly? With a sidetrip?

No, it does not.

## Part 2

1. Below are four graphs, each of which is defined by the sets of nodes, initial nodes, final nodes, edges, and defs and uses. Each graph also contains some test paths. Answer the following questions about each graph.



**Graph II.**

$N = \{1, 2, 3, 4, 5, 6\}$

$N_0 = \{1\}$

$N_f = \{6\}$

$E = \{(1,2), (2,3), (2,6), (3,4), (3,5), (4,5), (5,2)\}$

$def(1) = def(3) = use(3) = use(6) = \{x\}$

// Assume the use of x in 3 precedes the def

Test Paths:

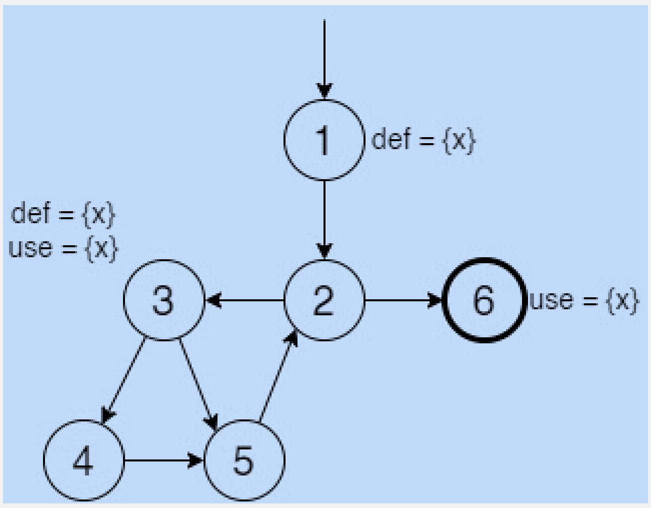
$t1 = [1, 2, 6]$

$t2 = [1, 2, 3, 4, 5, 2, 3, 5, 2, 6]$

$t3 = [1, 2, 3, 5, 2, 3, 4, 5, 2, 6]$

$t4 = [1, 2, 3, 5, 2, 6]$

a) Draw the graph.



b) List all of the du-paths with respect to x. (Note: Include all du-paths, even those that are subpaths of some other du-path).

(1,2,6)  (1,2,3)  (3,4,5,2,3)  (3,5,2,3)  (3,4,5,2,6)  (3,5,2,6)

c) Determine which du-paths each test path tours. Write them in a table with test paths in the first column and the du-paths they cover in the second column. For this part of the exercise, you should consider both direct touring and sidetrips.

I'll consider sidetrips, not de-tours

| | |
|---|---|
| 1, 2, 6 | (1,2,6) |
| 1, 2, 3, 4, 5, 2, 3, 5, 2, 6 | (1,2,3)<br>(1,2,6) with sidetrip<br>(3,4,5,2,3)<br>(3,5,2,6) |
| 1, 2, 3, 5, 2, 3, 4, 5, 2, 6 | (1,2,3)<br>(1,2,6) with sidetrip<br>(3,5,2,3)<br>(3,4,5,2,6) |
| 1, 2, 3, 5, 2, 6 | (2,5,2,6)<br>(1,2,3)<br>(1,2,6) with sidetrip |

d) List a minimal test set that satisfies all defs coverage with respect to x. (Direct tours only.) If possible, use the given test paths. If not, provide additional test paths to satisfy the criterion.

1, 2, 3, 4, 5, 2, 3, 5, 2, 6

e) List a minimal test set that satisfies all uses coverage with respect to x. (Direct tours only.) If possible, use the given test paths. If not, provide additional test paths to satisfy the criterion.

1, 2, 6

1, 2, 3, 4, 5, 2, 3, 5, 2, 6

f) List a minimal test set that satisfies all du-paths coverage with respect to x. (Direct tours only.) If possible, use the given test paths. If not, provide additional test paths to satisfy the criterion.

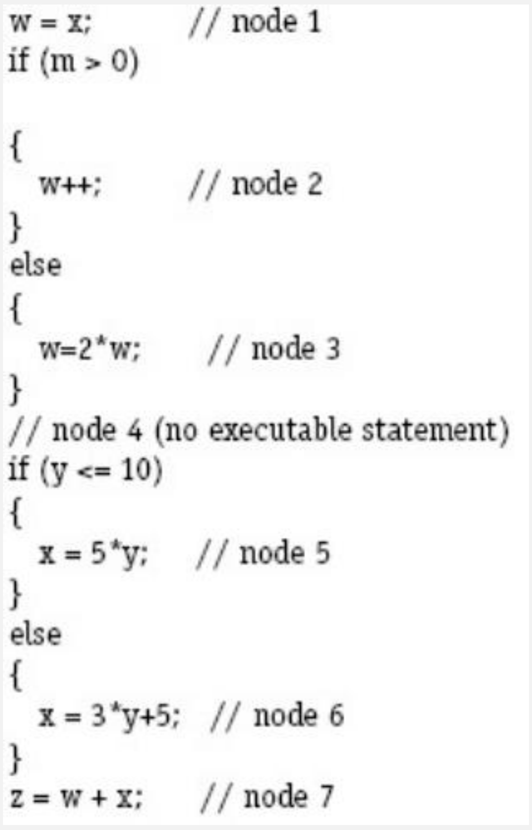
1, 2, 6

1, 2, 3, 4, 5, 2, 3, 5, 2, 6

1, 2, 3, 5, 2, 3, 4, 5, 2, 6

## Part 3

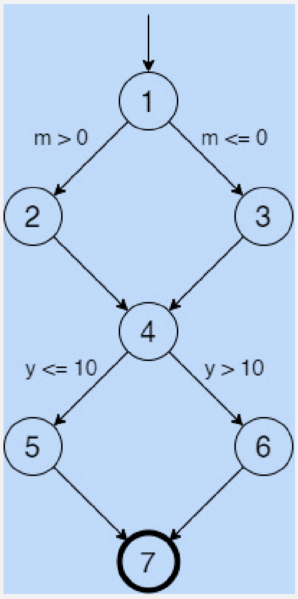
1. Use the following program fragment for questions a-e below.



```
w = x;          // node 1
if (m > 0)

{
   w++;         // node 2
}
else
{
   w=2*w;       // node 3
}
// node 4 (no executable statement)
if (y <= 10)
{
   x = 5*y;     // node 5
}
else
{
   x = 3*y+5;   // node 6
}
z = w + x;      // node 7
```

a) Draw a control flow graph for this program fragment. Use the node numbers given above.



b) Which nodes have defs for variable w?

1, 2, 3

c) Which nodes have uses for variable w?

2, 3, 7

d) Are there any du-paths with respect to variable w from node 1 to node 7? If not, explain why not. If any exist, show one.

No, because both nodes 2 and 3 have def for w, so the path from 1 to 7 is not def-clear

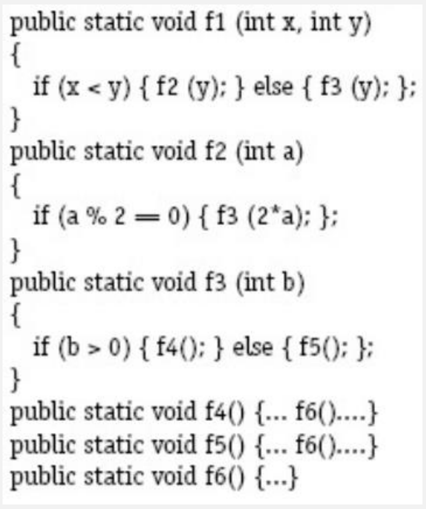
List all of the du-paths for variables w and x.

w: (2,4,5,7)  (2,4,6,7)  (3,4,5,7)  (3,4,6,7)

x: (5,7)  (6,7)

## Part 4

3. Use the following program fragment for questions a-e below.

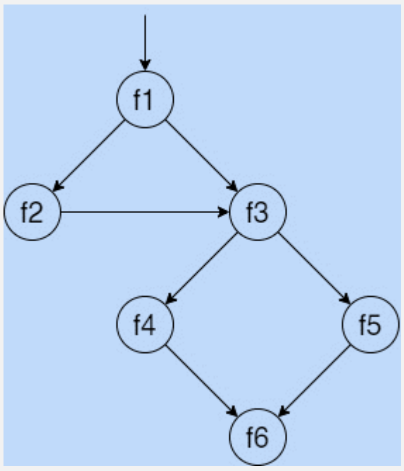


```
public static void f1 (int x, int y)
{
  if (x < y) { f2 (y); } else { f3 (y); };
}
public static void f2 (int a)
{
  if (a % 2 == 0) { f3 (2*a); };
}
public static void f3 (int b)
{
  if (b > 0) { f4(); } else { f5(); };
}
public static void f4() {... f6()....}
public static void f5() {... f6()....}
public static void f6() {...}
```

Use the following test inputs:

- t1 = f1 (0, 0)
- t2 = f1 (1, 1)
- t3 = f1 (0, 1)
- t4 = f1 (3, 2)
- t5 = f1 (3, 4)

a) Draw the call graph for this program fragment.



b) Give the path in the graph followed by each test.

t1: f1, f3, f5, f6

t2: f1, f3, f4, f6

t3: f1, f2

t4: f1, f3, f4, f6

t5: f1, f2, f3, f4, f6

c) Find a minimal test set that achieves Node Coverage.

[(3, 4), (0, 0)]

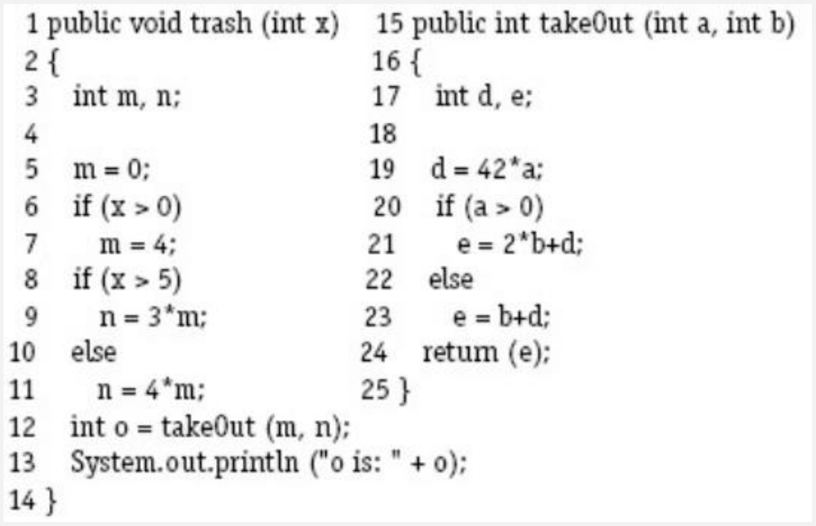
d) Find a minimal test set that achieves Edge Coverage.

[(3, 4), (0, 0)]

e) Given the prime paths in the graph. Which prime path is not covered by any of the tests above?

t5: f1, f2, f3, f5, f6

**4.** Use the following methods trash() and takeOut() to answer questions a-c.



```
1 public void trash (int x)     15 public int takeOut (int a, int b)
2 {                             16 {
3    int m, n;                  17    int d, e;
4                               18
5    m = 0;                     19    d = 42*a;
6    if (x > 0)                 20    if (a > 0)
7       m = 4;                  21       e = 2*b+d;
8    if (x > 5)                 22    else
9       n = 3*m;                23       e = b+d;
10   else                       24    return (e);
11      n = 4*m;                25 }
12   int o = takeOut (m, n);
13   System.out.println ("o is: " + o);
14 }
```

a) Give all call sites using the line numbers given.

12, 13

b) Give all pairs of last-def s and first-uses.

last-defs: (n,9)  (n,11)  (m,5)  (m,7)  (e,21)  (e,23)

first-uses: (o,13)  (a,19)  (b,21)  (b,23)  (x,6)  (x,8)

pairs:

(trash(), n, 9)  (takeOut(), b, 21)

(trash(), n, 9)  (takeOut(), b, 23)

(trash(), n, 11)  (takeOut(), b, 21)

(trash(), n, 11)  (takeOut(), b, 23)

(trash(), m, 5)  (takeOut(), a, 19)

(trash(), m, 7)  (takeOut(), a, 19)

(takeOut(), e, 21)  (trash(), o, 13)

(takeOut(), e, 23)  (trash(), o, 13)

c) Provide test inputs that satisfy all-coupling-uses (note that trash() only has one input).

x = -1  x = 2  x = 6

note that the coupling (trash(), n, 9) (takeOut(), b, 23) is infeasible