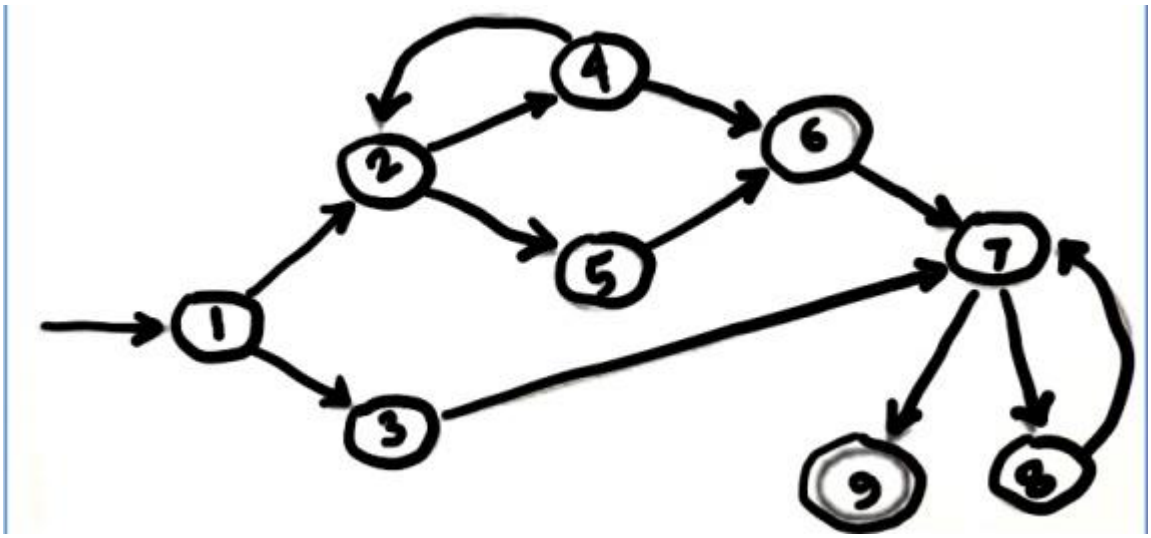
1) Define def-clear path and simple path. Draw the dataflow graph with proper annotations. Write down the edge pair coverage criteria of the derived graph. Determine the du-paths for the variables **isPat**, **rtnIndex**, **iSub**

```
public static int FindPatternIndex(String subject, String pattern){
final int NOTFOUND=-1;
int iSub=0, rtnIndex=NOTFOUND;
boolean isPat=false;
int subjectLen=subject.length();
int patternLen=pattern.length();
while(isPat==false && iSub+patternLen-1<subjectLen){
if(subject.charAt(iSub)==pattern.charAt(0)){
rtnIndex=iSub;
isPat=true;
for(int iPat=1; iPat<patternLen; iPat++){
if(subject.charAt(iSub+iPat)!=pattern.charAt(iPat)){
rtnIndex=NOTFOUND;
isPat=false;
break;
}
}
}
iSub++;
}
return rtnIndex;
}
```

2) For the expression, $\boldsymbol{E} = \boldsymbol{a} \wedge (\boldsymbol{b} \vee \boldsymbol{c})$ determine the CACC and RACC

3) Set Representation of Figure graph. Prime Path Coverage Criteria of Figure graph

Now write down the test requirements and test paths for each of the following criteria:
i. Edge Coverage
ii. Node Coverage
iii. Edge-pair Coverage



4)

Consider the following function.

```
public boolean findElement (List list, Object element)
// Effects: if list or element is null throw NullPointerException
// else return true if element is in the list, false otherwise
```

Now, give an example of partitioning scheme that will satisfy the following characteristic constraints for the above code snippet and highlight the criteria:
i) A block from one characteristic cannot be combined with a specific block from another.
ii) A block from one characteristic can ONLY BE combined with a specific block form another characteristic.

5)

Use the following characteristics and blocks for the questions below.

Characteristics	Block 1	Block 2	Block 3	Block 4
Value 1	< 0	0	> 0	
Value 2	< 0	0	> 0	
Operation	+	−	×	÷

i) Give test cases to satisfy the Each Choice criterion.
ii) Give test cases to satisfy the Base Choice criterion. Assume base choices are `Value 1 = > 0`, `Value 2 = > 0`, and `Operation = +`.
iii) How many tests are needed to satisfy the Pair-wise Coverage criterion?

6)

Consider the following information about a graph and answer each of the followings

N = {1, 2, 3, 4, 5, 6, 8}
N_o= {1}
N_f= {8}
E = {(1,2), (2,3), (2,8), (3,4), (3,5), (4,3), (5,6), (5,7), (6,7), (7,2) }
def(1) = def(4) = use(6) = use(8) = {x}

a) Draw the graph.

b) List all the du-paths with respect to x.

c) List a minimal test set that satisfies all uses coverage with respect to x.

d) List a minimal test set that satisfies all-du-paths coverage with respect to x.

7)

Consider the logic expression, $p = ((a < b) \vee D) \wedge (m >= n * o)$ and answer the followings:
i) List down the clauses
ii) Determine any test cases for clause coverage.

8)

Determine the CACC and RACC pairs of the clauses for the following logic expression:
$p = (\neg a \wedge \neg b) \vee (a \wedge \neg c) \vee (\neg a \wedge c)$

9)

Assume that, while doing ISP we found three characteristics $\{A, B, C\}$ and each of the characteristics
are partitioned into blocks of different sizes $\{(A1, A2), (B1, B2, B3), (C1, C2, C3, C4)\}$.
Now, answer each of the following questions:
i) How many test cases we will get for all combination coverage?
ii) How many test cases we will get for pair-wise coverage?
iii) How many test cases we will get for base choice coverage?

10)

Consider the following information about a graph and answer each of the followings

$N = \{1, 2, 3, 4, 5, 6, 7, 8\}$
$N_o = \{1\}$
$N_f = \{7\}$
$E = \{(1,5), (5,3), (3,4), (3,2), (4,3), (2,6), (6,8), (6,7)(8,6)\}$
$\text{def}(1) = \{x, y, z\}$; $\text{def}(4) = \{z\}$
$use(1) = use(4) = use(2) = use(8) = \{x\}$
$use(2) = use(7) = \{y\}$
$use(4) = use(2) = \{z\}$

a) Draw the graph.
b) List down the prime paths of the graph.
c) List all the du-paths with respect to variable x, y and z.
d) List down the minimal set of test paths for all-du-paths coverage with respect to variable x, y and z.

11)

A tester defined three characteristics based on the input parameter car: **Where Made**, **Energy Source**,and **Size**. The following partitionings for these characteristics have at least two mistakes. Identify them

Where Made		
North America	Europe	Asia
Energy Source		
Gas	Electric	Hybrid
Size		
2-Door	4-Door	Hatch-back

12)

Assume that, while doing ISP we found three characteristics $\{A, B, C, D\}$ and each of the characteristics are partitioned into following blocks: $\{(A1, A2, A3, A4), (B1, B2, B3), (C1, C2, C3, C4), (D1, D2)\}$.

Now, answer each of the following questions:

i) How many test cases we will get for all combination coverage?

ii) How many test cases we will get for each choice coverage?

iii) How many test cases we will get for base choice coverage?

iv) How many test cases we will get for 3-wise coverage?

13)

Suppose you have a predicate, $\boldsymbol{P} = (\boldsymbol{x} \vee \boldsymbol{y}) \wedge (\boldsymbol{p} \vee \boldsymbol{q})$. Now answer each of the followings:

i) What are the RACC pairs of clause x

ii) What are the RACC pairs of clause q

iii) What are the GACC pairs of clause y

iv) What are the GACC pairs of clause p

14)

Consider the logic expression, $\boldsymbol{P} = ((\boldsymbol{f} <= \boldsymbol{g}) \wedge (\boldsymbol{X} > \boldsymbol{0})) \vee (\boldsymbol{M} \wedge (\boldsymbol{e} < \boldsymbol{d} + \boldsymbol{c}))$ and answer the followings:

i) List down the clauses

ii) Write down any test case for clause coverage

15) Draw the control flow graph for each of the following code snippet.

```
x = 0;
while (x < y)
{
y = f (x, y);
if (y == 0)
{
break;
} else if (y < 0)
{
y = y*2;
continue;
}
x = x + 1;
}
```

code 1

```
/**
* Return index of node n at the
* first position it appears,
* -1 if it is not present
*/
public int indexOf (Node n)
{
if(n!=null)
{
for (int i=0; i < path.size(); i++)
if (path.get(i).equals(n))
return i;
}
return -1;
}
```

Code 2

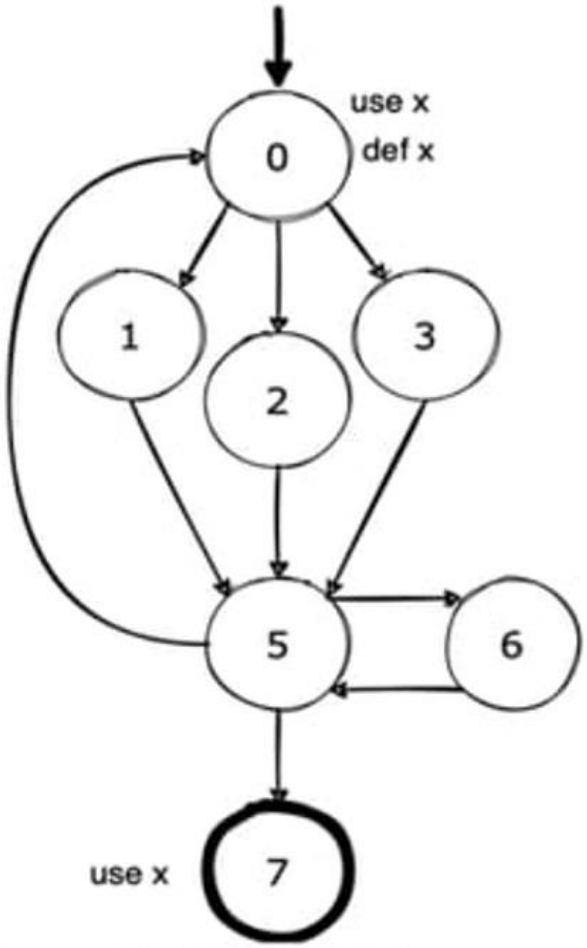
```
read ( c) ;
switch ( c )
{
case ‘N’:
z = 25;
case ‘Z’:
z += 20;
case ‘Y’:
x = 50;
break;
default:
x = 0;
break;
}
```

code 3

16)

For the logic expression, $\boldsymbol{p} = (\boldsymbol{x} \wedge \boldsymbol{y}) \vee (\boldsymbol{x} \wedge \neg \boldsymbol{y})$ show that only *clause* $\times$ determines the predicate.

17)



a) Write down the set representation of the above graph.

b) Write down the edge pair coverage of the above graph.

c) Write down all the du paths with respect to the variable *x* of the above graph.

d) Write down the prime paths and the corresponding test paths of the above graph.

18)

```
public static int countPrimes(int []inputs)
{
    int count=0, isprime;
    for(int i=0;i<arr.length;i++)
    {
       isprime=1;
       for(int j=2;j<inputs[i]*inputs[i];j++)
       {
          if(inputs[j]%i==0)
          {
            isprime=0;
            break;
          }
       }
       count+=isprime;
    }
    return count;
}
```

i) Define fault. If possible, identify the faults of the above code segment and correct it.

ii) Define failure. If possible, identify a test case that does not execute the fault.

iii) Define error. If possible, identify a test case that identifies the errors.