1) La transition coverage di un insieme di test cases (cioè sequenze di inputs) per uno state diagram è la percentuale di transizioni (archi nel grafo dello state diagram) percorsi almeno una volta.
Si consideri lo state diagram in figura
ed il seguente insieme di test cases:
Test case 1: act2 act2 act1 act2
Test case 2: act0 act2 act0
Test case 3: act0 act0 act0 act1 act1

Quale delle seguenti è la migliore stima della transition coverage per i test cases di cui sopra?

1.
Transition coverage: 70%
2.
Transition coverage: 30%
3.
Transition coverage: 40%
Risposta: The correct answer is: Transition coverage: 40%

2) Quale delle seguenti frasi meglio descrive il criterio di "requirements verifiability" che è parte della "requirements validation activity".

1.
Per ciascuna componente del sistema, dovremmo essere in grado di scrivere un insieme di test che può dimostrare che essa soddisfa tutti i requisiti.
2.
Per ciascun requisito, dovremmo essere in grado di scrivere un inseme di test che può dimostrare che il sistema sviluppato soddisfa il requisito considerato.
3.
Per ciascuna coppia di componenti, dovremmo essere in grado di scrivere un insieme di test che può dimostrare che l'interazione tra le componenti soddisfa tutti i requisiti di interfaccia.
Risposta: The correct answer is: Per ciascun requisito, dovremmo essere in grado di scrivere un inseme di test che può dimostrare che il sistema sviluppato soddisfa il requisito considerato.

3) Si consideri il seguente requisito:
RQ1: Dopo 20 unità di tempo dall'inizio dell'esecuzione la variabile x è sempre nell'intervallo [20, 30] .
Quale dei seguenti monitor meglio descrive il requisito RQ1 ?

1.
```
class Monitor

InputReal x;  // plant output
OutputBoolean y;

Boolean z;
initial equation
y = false;
equation
z = (time > 20) and (x >= 20) and (x <= 30) ;
algorithm
when edge(z) then
y := true;
end when;
end Monitor;
```
2.
```
class Monitor
```

```
InputReal x;  // plant output
OutputBoolean y;

Boolean z;
initial equation
y = false;
equation
z = (time > 20) or ((x >= 20) and (x <= 30)) ;
algorithm
when edge(z) then
y := true;
end when;
end Monitor;
```

3.

```
class Monitor

InputReal x;  // plant output
OutputBoolean y;

Boolean z;
initial equation
y = false;
equation
z = (time > 20) and ((x >= 30) or (x <= 20)) ;
algorithm
when edge(z) then
y := true;
end when;
end Monitor;
```

Risposta: The correct answer is: class Monitor

```
InputReal x;  // plant output
OutputBoolean y;

Boolean z;
initial equation
y = false;
equation
z = (time > 20) and ((x >= 30) or (x <= 20)) ;
algorithm
when edge(z) then
y := true;
end when;
end Monitor;
```

4) La transition coverage di un insieme di test cases (cioè sequenze di inputs) per uno state diagram è la percentuale di transizioni (archi nel grafo dello state diagram) percorsi almeno una volta.
Si consideri lo state diagram in figura
ed il seguente insieme di test cases:
Test case 1: act1 act0 act1 act0 act2
Test case 2: act0 act2 act2 act0 act1
Test case 3: act0 act0

Quale delle seguenti è la migliore stima della transition coverage per i test cases di cui sopra?

1.
Transition coverage: 35%
2.
Transition coverage: 80%
3.
Transition coverage: 60%
Risposta: The correct answer is: Transition coverage: 35%

5) Una Condition è una proposizione booleana, cioè una espressione con valore booleano che non può essere decomposta in espressioni boolean più semplici. Ad esempio, (x + y <= 3) è una condition.

Una Decision è una espressione booleana composta da conditions e zero o più operatori booleani. Ad esempio, sono decisions:
(x + y <= 3)
((x + y <= 3) || (x - y > 7))
Un insieme di test cases T soddisfa il criterio di Condition/Decision coverage se tutte le seguenti condizioni sono soddisfatte:

1) Ciascun punto di entrata ed uscita nel programma è eseguito in almeno un test;
2) Per ogni decision d nel programma, per ogni condition c in d, esiste un test in T in cui c è true ed un test in T in cui c è false.
3) Per ogni decision d nel programma, esiste un test in T in cui d è true ed un test in T in cui d è false.

Si consideri la seguente funzione:

```
int f(int a, int b, int c)
{    if ( (a  + b - 6 >= 0) && (b - c - 1 <= 0) )
        return (1);    // punto di uscita 1
     else if ((b - c - 1 <= 0)  || (b + c - 5 >= 0))
         then return (2);   // punto di uscita 2
         else return (3);   // punto di uscita 3
}
```

Quale dei seguenti test set soddisfa il criterio della Condition/Decision coverage ?

1.
(a = 6, b = 0, c = 1), (a = 0, b = 5, c = 0), (a = 0, b = 3, c = 0).
2.
(a = 6, b = 0, c = 1), (a = 0, b = 5, c = 0), (a = 0, b = 3, c = 2).
3.
(a = 5, b = 0, c = 1), (a = 0, b = 5, c = 0), (a = 0, b = 3, c = 0).
Risposta: The correct answer is:
(a = 6, b = 0, c = 1), (a = 0, b = 5, c = 0), (a = 0, b = 3, c = 0).

6) Il rischio R può essere calcolato come R = P*C, dove P è la probabilità dell'evento avverso (software failure nel nostro contesto) e C è il costo dell'occorrenza dell'evento avverso.
Assumiamo che la probabilità P sia legata al costo di sviluppo S dalla formula
P = 10^{(-b*S)} (cioè 10 elevato alla (-b*S))
dove b è una opportuna costante note da dati storici aziendali. Si assuma che b = 0.0001, C = 1000000, ed il rischio ammesso è R = 1000. Quale dei seguenti valori meglio approssima il costo S per lo sviluppo del software in questione.

1.
500000 EUR
2.
300000 EUR
3.

700000 EUR
Risposta: The correct answer is: 300000 EUR

7) Un processo software può essere rappesentato con uno state diagram in cui gli stati rappresentano le fasi (e loro iterazioni) del processo software e gli archi le transizioni da una fase all'altra. Gli archi sono etichettati con le probabilità della transizione e gli stati sono etichettati con il costo per lasciare lo stato.
Ad esempio lo state diagram in figura
Rappresenta un processo software con 2 fasi F1 ed F2. F1 ha costo 10000 EUR ed F2 ha costo 1000 EUR. F1 ha una probabilita dello 0.4 di dover essere ripetuta (a causa di errori) ed F2 ha una probabilità 0.3 di dover essere ripetuta (a causa di errori).
Uno scenario è una sequenza di stati.
Qual'e' la probabilità dello scenario: 1, 2, 3? In altri terminti, qual'è la probabilità che non sia necessario ripetere la prima fase (ma non la seconda) ?

1.
0.12
2.
0.28
3.
0.42
Risposta: The correct answer is: 0.28

8) Si consideri il seguente requisito:
RQ: Dopo 10 unità di tempo dall'inizio dell'esecuzione vale la seguente proprietà: se la variabile x è nell'intervallo [10, 20] allora la variabile y è compresa tra il 50% di x ed il 70% di x.
Quale dei seguenti monitor meglio descrive il requisito RQ ?

1.
class Monitor

InputReal x, y; // plant output
OutputBoolean wy;

Boolean wz;
initial equation
wy = false;
equation
wz = (time > 10) and (x >= 10) and (x <= 20) and (y >= 0.5*x) and (y <= 0.7*x) ;
algorithm
when edge(wz) then
wy := true;
end when;
end Monitor;
2.
class Monitor

InputReal x, y; // plant output
OutputBoolean wy;

Boolean wz;
initial equation
wy = false;
equation
wz = (time > 10) and (x >= 10) and (x <= 20) and ((y < 0.5*x) or (y > 0.7*x)) ;

algorithm
when edge(wz) then
wy := true;
end when;
end Monitor;

3.

class Monitor

InputReal x, y; // plant output
OutputBoolean wy;

Boolean wz;
initial equation
wy = false;
equation
wz = (time > 10) and ((x < 10) or (x > 20)) and ((y < 0.5*x) or (y > 0.7*x)) ;
algorithm
when edge(wz) then
wy := true;
end when;
end Monitor;

Risposta: The correct answer is: class Monitor

InputReal x, y; // plant output
OutputBoolean wy;

Boolean wz;
initial equation
wy = false;
equation
wz = (time > 10) and (x >= 10) and (x <= 20) and ((y < 0.5*x) or (y > 0.7*x)) ;
algorithm
when edge(wz) then
wy := true;
end when;
end Monitor;

9) La state coverage di un insieme di test cases (cioè sequenze di inputs) per uno state diagram è la percentuale di stati (inclusi START ed END) raggiunti almeno una volta.
Si consideri lo state diagram in figura
Si consideri il seguente insieme di test cases:
Test case 1: act1 act0 act1 act0 act2
Test case 2: act0 act2 act2 act0 act1
Test case 3: act0 act0
Quale delle seguenti è la migliore stima della state coverage per i test cases di cui sopra

1.

State coverage: 50%

2.

State coverage: 80%

3.

State coverage: 100%

Risposta: The correct answer is: State coverage: 100%

10) Il partition coverage di un insieme di test cases è la percentuale di elementi della partition inclusi nei test cases. La partition è una partizione finita dell'insieme di input della funzione che si sta testando.
Si consideri la seguente funzione C:
int f1(int x) { return (2*x); }
Si vuole testare la funzione f1(). A tal fine l'insieme degli interi viene partizionato come segue:
{(-inf, -11], [-10, -1], {0}, [1, 50], [51, +inf)}
Si consideri il seguente insieme di test cases:
{x=-20, x= 10, x=60}
Quale delle seguenti è la partition coverage conseguita?

1.
60%
2.
80%
3.
100%
Risposta: The correct answer is: 60%

11) Un processo software può essere rappesentato con uno state diagram in cui gli stati rappresentano le fasi (e loro iterazioni) del prcoesso software e gli archi le transizioni da una fase all'altra. Gli archi sono etichettati con le probabilità della transizione e gli stati sono etichettati con il costo per lasciare lo stato.
Ad esempio lo state diagram in figura
Rappresenta un processo software con 2 fasi F1 ed F2. F1 ha costo 10000 EUR ed F2 ha costo 1000 EUR. F1 ha una probabilita dello 0.3 di dover essere ripetuta (a causa di errori) ed F2 ha una probabilità 0.2 di dover essere ripetuta (a causa di errori).
Uno scenario è una sequenza di stati.
Qual'e' la probabilità dello scenario: 1, 3 ? In altri terminti, qual'è la probabilità che non sia necessario ripetere nessuna fase?

1.
0.56
2.
0.14
3.
0.24
Risposta: The correct answer is: 0.56

12) Quale delle seguenti frasi meglio descrive l'obiettivo del "validity check" che è parte della "requirements validation activity".

1.
Assicurarsi che un sistema che soddisfa i requisiti risolve il problema del "customer".
2.
Assicurarsi che non ci siano requisiti in conflitto con altri requisiti.
3.
Assicurarsi che i requisiti funzionali descrivano tutte le funzionalità del sistema.
Risposta: The correct answer is: Assicurarsi che un sistema che soddisfa i requisiti risolve il problema del "customer".

13) Un processo software può essere rappesentato con uno state diagram in cui gli stati rappresentano le fasi (e loro iterazioni) del prcoesso software e gli archi le transizioni da una fase all'altra. Gli archi sono etichettati con le probabilità della transizione e gli stati sono etichettati con il costo per lasciare lo stato.
Ad esempio lo state diagram in figura
Rappresenta un processo software con 2 fasi F1 ed F2. F1 ha costo 10000 EUR ed F2 ha costo 1000 EUR. F1 ha una probabilita dello 0.4 di dover essere ripetuta (a causa di errori) ed F2 ha una probabilità 0.2 di dover essere ripetuta (a causa di errori).
Uno scenario è una sequenza di stati.
Qual'e' la probabilità dello scenario: 1, 3, 4? In altri terminti, qual'è la probabilità che non sia necessario ripetere la seconda fase (ma non la prima)

?

1.
0.08
2.
0.32
3.
0.12
Risposta: The correct answer is: 0.12

14) Un processo software può essere rappesentato con uno state diagram in cui gli stati rappresentano le fasi (e loro iterazioni) del prcoesso software e gli archi le transizioni da una fase all'altra. Gli archi sono etichettati con le probabilità della transizione e gli stati sono etichettati con il costo per lasciare lo stato.
Ad esempio lo state diagram in figura
Rappresenta un processo software con 2 fasi F1 ed F2. F1 ha costo 10000 EUR ed F2 ha costo 1000 EUR. F1 ha una probabilita dello 0.3 di dover essere ripetuta (a causa di errori) ed F2 ha una probabilità 0.1 di dover essere ripetuta (a causa di errori).
Uno scenario è una sequenza di stati.
Qual'e' la probabilità dello scenario: 1, 2, 3, 4 ? In altri terminti, qual'è la probabilità che sia necessario ripetere sia la fase 1 che la fase 2 ?

1.
0.07
2.
0.03
3.
0.27
Risposta: The correct answer is: 0.03

15) Si consideri il seguente requisito:
RQ: Dopo 60 unità di tempo dall'inizio dell'esecuzione vale la seguente proprietà:
se 10 unità di tempo nel passato era stata richiesta una risorsa (variabile x positiva) allora ora è concesso l'accesso alla risorsa (variabile y positiva)
Tenendo presente che, al tempo time, delay(z, w) ritorna 0 se time < w e ritorna il valore che z aveva al tempo (time - w), se time >= w.
Quale dei seguenti monitor meglio descrive il requisito RQ ?

1.
```
class Monitor
InputReal x, y;
OutputBoolean wy;
Boolean wz;
initial equation
wy = false;
equation
wz = (time > 60) and (delay(x, 10) > 0) and (y <= 0);
algorithm
when edge(wz) then
wy := true;
end when;
end Monitor;
```
2.
```
class Monitor
InputReal x, y;
OutputBoolean wy;
Boolean wz;
initial equation
```

```
wy = false;
equation
wz = (time > 60) and (delay(x, 10) > 0) and (y > 0);
algorithm
when edge(wz) then
wy := true;
end when;
end Monitor;
```

3.

```
class Monitor
InputReal x, y;
OutputBoolean wy;
Boolean wz;
initial equation
wy = false;
equation
wz = (time > 60) or (delay(x, 10) > 0) or  (y <= 0);

algorithm
when edge(wz) then
wy := true;
end when;
end Monitor;
```

Risposta: The correct answer is: class Monitor

```
InputReal x, y;
OutputBoolean wy;
Boolean wz;
initial equation
wy = false;
equation
wz = (time > 60) and (delay(x, 10) > 0) and (y <= 0);
algorithm
when edge(wz) then
wy := true;
end when;
end Monitor;
```

16) La state coverage di un insieme di test cases (cioè sequenze di inputs) per uno state diagram è la percentuale di stati (inclusi START ed END) raggiunti almeno una volta.
Si consideri lo state diagram in figura
Si consideri il seguente insieme di test cases:
Test case 1: act1 act2 act1 act2 act2
Test case 2: act2 act0
Test case 3: act0 act0 act0
Quale delle seguenti è la migliore stima della state coverage per i test cases di cui sopra

1.

State coverage: 90%

2.

State coverage: 60%

3.

State coverage: 70%

Risposta: The correct answer is: State coverage: 90%

17) Si consideri il seguente modello Modelica. Quale dei seguenti UML state diagram lo rappresenta correttamente ?

```
block FSA  //  Finite State Automaton

/* connector declarations outside this block:
connector InputInteger = input Integer;
connector OutputInteger = output Integer;
*/

InputInteger u; // external input
OutputInteger x; // state
parameter Real T = 1;

algorithm

when initial() then
x := 0;

elsewhen sample(0,T) then

if (pre(x) == 1) and (pre(u) == 0) then x := 3;
elseif (pre(x) == 1) and (pre(u) == 1) then x := 3;
elseif (pre(x) == 1) and (pre(u) == 2) then x := 2;
elseif (pre(x) == 2) and (pre(u) == 0) then x := 0;
elseif (pre(x) == 2) and (pre(u) == 2) then x := 0;
elseif (pre(x) == 3) and (pre(u) == 0) then x := 2;
elseif (pre(x) == 4) and (pre(u) == 0) then x := 3;
elseif (pre(x) == 4) and (pre(u) == 2) then x := 2;
else x := pre(x); // default
end if;

end when;
end FSA;
```

18) Si consideri il monitor seguente che ritorna true appena i requisiti per il sistema monitorato sono violati.

```
block Monitor
input Real x;
output Boolean y;
Boolean w;
initial equation
y = false;
equation
w = ((x < 0) or (x > 5));
algorithm
when edge(w) then
y := true;
end when;
end Monitor;
```

Quale delle seguenti affermazioni meglio descrive il requisito monitorato.

1. La variabile x è nell'intervallo [0, 5].
2. La variable x è minore di 0.

3.
La variabile x è fuori dall'intervallo [0, 5].
Risposta: The correct answer is: La variabile x è nell'intervallo [0, 5].

19) La transition coverage di un insieme di test cases (cioè sequenze di inputs) per uno state diagram è la percentuale di transizioni (archi nel grafo dello state diagram) percorsi almeno una volta.
Si consideri lo state diagram in figura
ed il seguente insieme di test cases:
Test case 1: act2 act1
Test case 2: act1 act0 act1 act0 act2
Test case 3: act0 act2 act2 act1
Quale delle seguenti è la migliore stima della transition coverage per i test cases di cui sopra?

1.
Transition coverage: 40%
2.
Transition coverage: 30%
3.
Transition coverage: 80%
Risposta: The correct answer is: Transition coverage: 40%

20) Si consideri il seguente modello Modelica. Quale dei seguenti UML state diagram lo rappresenta correttamente ?

```
block FSA  //  Finite State Automaton

/* connector declarations outside this block:
connector InputInteger = input Integer;
connector OutputInteger = output Integer;
*/

InputInteger u; // external input
OutputInteger x; // state
parameter Real T = 1;

algorithm

when initial() then
x := 0;

elsewhen sample(0,T) then

if (pre(x) == 0) and (pre(u) == 0) then x := 1;
elseif (pre(x) == 0) and (pre(u) == 1) then x := 1;
elseif (pre(x) == 0) and (pre(u) == 2) then x := 1;
elseif (pre(x) == 1) and (pre(u) == 1) then x := 4;
elseif (pre(x) == 1) and (pre(u) == 2) then x := 0;
elseif (pre(x) == 2) and (pre(u) == 0) then x := 4;
elseif (pre(x) == 2) and (pre(u) == 1) then x := 1;
elseif (pre(x) == 3) and (pre(u) == 0) then x := 4;
elseif (pre(x) == 4) and (pre(u) == 0) then x := 2;
elseif (pre(x) == 4) and (pre(u) == 1) then x := 0;
elseif (pre(x) == 4) and (pre(u) == 2) then x := 2;
else x := pre(x); // default
end if;
```

```
end when;
end FSA;
```

21) Quale delle seguenti frasi meglio descrive l'obiettivo del "check di completezza" che è parte della "requirements validation activity".

1.
Assicurarsi che i requisisti descrivano tutte le funzionalità e vincoli (e.g., security, performance) del sistema desiderato dal customer.
2.
Assicurarsi che per ogni requisito sia stato implementato nel sistema.
3.
Assicurarsi che i requisiti funzionali descrivano tutte le funzionalità del sistema.
Risposta: The correct answer is: Assicurarsi che i requisisti descrivano tutte le funzionalità e vincoli (e.g., security, performance) del sistema desiderato dal customer.

22) Quale dei seguenti modelli Modelica rappresenta lo state diagram in figura ?

2.

```
block FSA  //  Finite State Automaton

/* connector declarations outside this block:
connector InputInteger = input Integer;
connector OutputInteger = output Integer;
*/

InputInteger u; // external input
OutputInteger x; // state
parameter Real T = 1;

algorithm

when initial() then
x := 0;

elsewhen sample(0,T) then

if (pre(x) == 0) and (pre(u) == 1) then x := 4;
elseif (pre(x) == 0) and (pre(u) == 2) then x := 4;
elseif (pre(x) == 1) and (pre(u) == 2) then x := 4;
elseif (pre(x) == 2) and (pre(u) == 0) then x := 0;
elseif (pre(x) == 2) and (pre(u) == 1) then x := 3;
elseif (pre(x) == 2) and (pre(u) == 2) then x := 1;
elseif (pre(x) == 3) and (pre(u) == 0) then x := 0;
elseif (pre(x) == 3) and (pre(u) == 1) then x := 2;
elseif (pre(x) == 3) and (pre(u) == 2) then x := 2;
elseif (pre(x) == 4) and (pre(u) == 0) then x := 1;
elseif (pre(x) == 4) and (pre(u) == 1) then x := 2;
elseif (pre(x) == 4) and (pre(u) == 2) then x := 1;
else x := pre(x); // default
end if;

end when;
end FSA;
```

3.

```
block FSA  //  Finite State Automaton

/* connector declarations outside this block:
connector InputInteger = input Integer;
connector OutputInteger = output Integer;
*/

InputInteger u; // external input
OutputInteger x; // state
parameter Real T = 1;

algorithm

when initial() then
x := 0;

elsewhen sample(0,T) then

if (pre(x) == 0) and (pre(u) == 0) then x := 3;
elseif (pre(x) == 0) and (pre(u) == 1) then x := 1;
elseif (pre(x) == 2) and (pre(u) == 0) then x := 0;
elseif (pre(x) == 2) and (pre(u) == 2) then x := 0;
elseif (pre(x) == 4) and (pre(u) == 0) then x := 0;
else x := pre(x); // default
end if;

end when;
end FSA;
```

23) Quale delle seguenti frasi meglio descrive l'obiettivo del "check di consistenza" che è parte della "requirements validation activity".

1.
Assicurarsi che per ogni requisito esista un insieme di test che lo possa verificare.
2.
Assicurarsi che i requisiti funzionali descrivano tutte le funzionalità del sistema.
3.
Assicurarsi che non ci siano requisiti in conflitto con altri requisiti.

Risposta: The correct answer is: Assicurarsi che non ci siano requisiti in conflitto con altri requisiti.

24) Il branch coverage di un insieme di test cases è la percentuale di branch del programma che sono attraversati da almeno un test case.
Si consideri la seguente funzione C:

```
----------
int f(int x, int y)  {
 if (x - y <= 0)   { if (x + y - 2>= 0)  return (1); else return (2); }
  else {if (2*x + y - 1>= 0)  return (3); else return (4); }
 }  /* f()  */
```

Si considerino i seguenti test cases: {x=1, y=1}, {x=0, y=0}, {x=1, y=0}, {x=0, y=-1}.
Quale delle seguenti è la branch coverage conseguita?

1.
100%
2.
50%

3.
80%
Risposta: The correct answer is: 100%

25) Si consideri la Markov chain in figura con stato iniziale 0 e p in (0, 1). Quale delle seguenti formule calcola il valore atteso del numero di transizioni necessarie per lasciare lo stato 0.

1.
(1 - p)/p
2.
1/(1 - p)
3.
1/(p*(1 - p))
Risposta: The correct answer is: 1/(1 - p)

26) Si consideri il processo software con due fasi (0 ed 1) rappresentato con la Markov chain in figura. Lo stato iniziale 0 e p è in (0, 1). Il tempo necessario per completare la fase x è time(x). La fase 0 è la fase di design, che ha probabilità p di dover essere ripetuta causa errori. La fase 1 rappreenta il completamento del processo software, e quindi time(1) = 0.
Il tempo di una istanza del processo software descritto sopra è la somma dei tempi degli stati (fasi) attraversati (tenendo presente che si parte sempre dallo stato 0.
Quindi il costo Time(X) della sequenza di stati X = x(0), x(1), x(2), ... è Time(X) = time(x(0)) + time(x(1)) + time(x(2)) + ...
Ad esempio se X = 0, 1 abbiamo Time(X) = time(0) + time(1) = time(0) (poichè time(1) = 0).
Quale delle seguenti formule calcola il valore atteso del costo per completare il processo software di cui sopra

1.
time(0)*(1 - p)/p
2.
time(0)/(p*(1 - p))
3.
time(0)/(1 - p)
Risposta: The correct answer is: time(0)/(1 - p)

27) Quali delle seguenti attività è parte del processo di validazione dei requisiti ?

1.
Accertarsi che i requisiti definiscano un sistema che risolve il problema che l'utente pianifica di risolvere.
2.
Accertarsi che il sistema soddisfi i requisiti dati.
3.
Accertarsi che l'architettura del sistema soddisfi i requisiti dati.
Risposta: The correct answer is: Accertarsi che i requisiti definiscano un sistema che risolve il problema che l'utente pianifica di risolvere.

28) Quale dei seguenti modelli Modelica rappresenta lo state diagram in figura ?

3.
block FSA // Finite State Automaton

/* connector declarations outside this block:
connector InputInteger = input Integer;

```
connector OutputInteger = output Integer;
*/

InputInteger u; // external input
OutputInteger x; // state
parameter Real T = 1;

algorithm

when initial() then
x := 0;

elsewhen sample(0,T) then

if (pre(x) == 0) and (pre(u) == 0) then x := 2;
elseif (pre(x) == 0) and (pre(u) == 1) then x := 4;
elseif (pre(x) == 0) and (pre(u) == 2) then x := 1;
elseif (pre(x) == 1) and (pre(u) == 0) then x := 3;
elseif (pre(x) == 1) and (pre(u) == 2) then x := 3;
elseif (pre(x) == 2) and (pre(u) == 0) then x := 3;
elseif (pre(x) == 2) and (pre(u) == 1) then x := 0;
elseif (pre(x) == 2) and (pre(u) == 2) then x := 3;
elseif (pre(x) == 4) and (pre(u) == 2) then x := 1;
else x := pre(x); // default
end if;

end when;
end FSA;
```

29) Si consideri il seguente requisito:
RQ: Dopo 50 unità di tempo dall'inizio dell'esecuzione vale la seguente proprietà:
se la variabile x è minore del 60% della variabile y allora la somma di x ed y è maggiore del 30% della variabile z
Quale dei seguenti monitor meglio descrive il requisito RQ ?

1.

```
class Monitor

InputReal x, y, z;  // plant output
OutputBoolean wy;
Boolean wz;
initial equation
wy = false;
equation
wz = (time > 50) and (x >= 0.6*y) and (x + y <= 0.3*z);
algorithm
when edge(wz) then
wy := true;
end when;
end Monitor;
```

2.

```
class Monitor

InputReal x, y, z;  // plant output
OutputBoolean wy;
```

```
Boolean wz;
initial equation
wy = false;
equation
wz = (time > 50) and (x < 0.6*y) and (x + y <= 0.3*z);
algorithm
when edge(wz) then
wy := true;
end when;
end Monitor;
```

3.

```
class Monitor

InputReal x, y, z;  // plant output
OutputBoolean wy;
Boolean wz;
initial equation
wy = false;
equation
wz = (time > 50) and (x < 0.6*y) and (x + y > 0.3*z);
algorithm
when edge(wz) then
wy := true;
end when;
end Monitor;
```

Risposta: The correct answer is: class Monitor

```
InputReal x, y, z;  // plant output
OutputBoolean wy;
Boolean wz;
initial equation
wy = false;
equation
wz = (time > 50) and (x < 0.6*y) and (x + y <= 0.3*z);
algorithm
when edge(wz) then
wy := true;
end when;
end Monitor;
```

30) Il branch coverage di un insieme di test cases è la percentuale di branch del programma che sono attraversati da almeno un test case.
Si consideri la seguente funzione C:

```
----------
int f(int x, int y)  {
 if (x - y - 6 <= 0)   { if (x + y - 3 >= 0)  return (1); else return (2); }
  else {if (x + 2*y -15 >= 0)  return (3); else return (4); }
 }  /* f()  */
```

Quale dei seguenti test sets consegue una branch coverage del 100% ?

1.
Test set: {x=3, y=6}, {x=0, y=0}, {x=15, y=0}, {x=9, y=0}.
2.

Test set: {x=3, y=6}, {x=0, y=0}, {x=15, y=0}, {x=10, y=3}.

3.

Test set: {x=3, y=6}, {x=2, y=1}, {x=15, y=0}, {x=9, y=0}.

Risposta: The correct answer is: Test set: {x=3, y=6}, {x=0, y=0}, {x=15, y=0}, {x=9, y=0}.

31) Si consideri il seguente requisito:

RQ: Durante l'esecuzione del programma (cioè per tutti gli istanti di tempo positivi) la variabile x è sempre nell'intervallo [0, 5].

Quale dei seguenti monitor meglio descrive il requisito RQ ?

1.

```
class Monitor

InputReal x;  // plant output
OutputBoolean y;

Boolean z;
initial equation
y = false;
equation
z = (time > 0) and (x > 0) and (x < 5);
algorithm
when edge(z) then
y := true;
end when;
end Monitor;
```

2.

```
class Monitor

InputReal x;  // plant output
OutputBoolean y;

Boolean z;
initial equation
y = false;
equation
z = (time > 0) and ((x > 5) or (x < 0));
algorithm
when edge(z) then
y := true;
end when;
end Monitor;
```

3.

```
class Monitor

InputReal x;  // plant output
OutputBoolean y;

Boolean z;
initial equation
y = false;
equation
z =  (time > 0) and ((x > 0) or (x < 5));
algorithm
```

when edge(z) then
y := true;
end when;
end Monitor;
Risposta: The correct answer is: class Monitor

InputReal x; // plant output
OutputBoolean y;

Boolean z;
initial equation
y = false;
equation
z = (time > 0) and ((x > 5) or (x < 0));
algorithm
when edge(z) then
y := true;
end when;
end Monitor;

32) Il partition coverage di un insieme di test cases è la percentuale di elementi della partition inclusi nei test cases. La partition è una partizione finita dell'insieme di input della funzione che si sta testando.
Si consideri la seguente funzione C:
int f1(int x) { return (x + 7); }
Si vuole testare la funzione f1(). A tal fine l'insieme degli interi viene partizionato come segue:
{(-inf, -101], [-100, -1], {0}, [1, 500], [501, +inf)}
Quale dei seguenti test cases consegue una partition coverage del 100% ?

1.
{x = -200, x = -150, x = 0, x = 100, x = 700}
2.
{x = -150, x = -40, x = 0, x = 200, x = 600}
3.
{x = -200, x = -50, x = 0, x = 100, x = 500}
Risposta: The correct answer is: {x = -150, x = -40, x = 0, x = 200, x = 600}

33) La transition coverage di un insieme di test cases (cioè sequenze di inputs) per uno state diagram è la percentuale di transizioni (archi nel grafo dello state diagram) percorsi almeno una volta.
Si consideri lo state diagram in figura
ed il seguente insieme di test cases:
Test case 1: act2 act2 act1 act2 act1
Test case 2: act1 act0 act2
Test case 3: act2 act1 act0
Quale delle seguenti è la migliore stima della transition coverage per i test cases di cui sopra?

1.
Transition coverage: 30%
2.
Transition coverage: 80%
3.
Transition coverage: 50%
Risposta: The correct answer is: Transition coverage: 50%

34) Si consideri il seguente modello Modelica. Quale dei seguenti UML state diagram lo rappresenta correttamente ?

```
block FSA  //  Finite State Automaton

/* connector declarations outside this block:
connector InputInteger = input Integer;
connector OutputInteger = output Integer;
*/

InputInteger u; // external input
OutputInteger x; // state
parameter Real T = 1;

algorithm

when initial() then
x := 0;

elsewhen sample(0,T) then

if (pre(x) == 0) and (pre(u) == 0) then x := 3;
elseif (pre(x) == 1) and (pre(u) == 1) then x := 2;
elseif (pre(x) == 1) and (pre(u) == 2) then x := 4;
elseif (pre(x) == 2) and (pre(u) == 1) then x := 4;
elseif (pre(x) == 2) and (pre(u) == 2) then x := 3;
elseif (pre(x) == 3) and (pre(u) == 0) then x := 1;
elseif (pre(x) == 3) and (pre(u) == 1) then x := 2;
elseif (pre(x) == 3) and (pre(u) == 2) then x := 1;
elseif (pre(x) == 4) and (pre(u) == 0) then x := 3;
elseif (pre(x) == 4) and (pre(u) == 1) then x := 2;
elseif (pre(x) == 4) and (pre(u) == 2) then x := 0;
else x := pre(x); // default
end if;

end when;
end FSA;
```

35) Si consideri il monitor seguente che ritorna true appena il sistema viola il requisito monitorato.

```
block Monitor
input Real x;
output Boolean y;
Boolean w;
initial equation
y = false;
equation
w = ((x < 1) or (x > 4)) and ((x < 15) or (x > 20));
algorithm
when edge(w) then
y := true;
end when;
end Monitor;
```

Quale delle seguenti affermazioni meglio descrive il requisito monitorato?

1.

La variabile x è nell'intervallo [1, 4] e fuori dall'intervallo [15, 20].

2.

La variabile x è nell'intervallo [1, 4] oppure nell'intervallo [15, 20].

3.

La variabile x è fuori dall'intervallo [1, 4] e fuori dall'intervallo [15, 20].

Risposta: The correct answer is: La variabile x è nell'intervallo [1, 4] oppure nell'intervallo [15, 20].

36) La state coverage di un insieme di test cases (cioè sequenze di inputs) per uno state diagram è la percentuale di stati (inclusi START ed END) raggiunti almeno una volta.

Si consideri lo state diagram in figura

Si consideri il seguente insieme di test cases:

Test case 1: act1 act0 act0

Test case 2: act2 act0 act1

Test case 3: act0 act0 act0

Quale delle seguenti è la migliore stima della state coverage per i test cases di cui sopra

1.

State coverage: 80%

2.

State coverage: 60%

3.

State coverage: 25%

Risposta: The correct answer is: State coverage: 25%

37) Si consideri il processo software con due fasi (0 ed 1) rappresentato con la Markov chain in figura. Lo stato iniziale 0 e p è in (0, 1). Il costo dello stato (fase) x è c(x). La fase 0 è la fase di design, che ha probabilità p di dover essere ripetuta causa errori. La fase 1 rappreenta il completamento del processo software, e quindi c(1) = 0.

Il costo di una istanza del processo software descritto sopra è la somma dei costi degli stati attraversati (tenendo presente che si parte sempre dallo stato 0.

Quindi il costo C(X) della sequenza di stati X = x(0), x(1), x(2), ... è C(X) = c(x(0)) + c(x(1)) + c(x(2)) + ...

Ad esempio se X = 0, 1 abbiamo C(X) = c(0) + c(1) = c(0) (poichè c(1) = 0).

Quale delle seguenti formule calcola il valore atteso del costo per completare il processo software di cui sopra

1.

c(0)/(1 - p)

2.

c(0)*(1 - p)/p

3.

c(0)/(p*(1 - p))

Risposta: The correct answer is: c(0)/(1 - p)

38) Pre-condizioni, invarianti e post-condizioni di un programma possono essere definiti usando la macro del C assert() (in <assert.h>). In particolare, assert(expre) non fa nulla se l'espressione expre vale TRUE (cioè non è 0), stampa un messaggio di errore su stderr e abortisce l'esecuzione del programma altrimenti.

Si consideri la funzione C

int f(int x, int y) { ..... }

Quale delle seguenti assert esprime la pre-condizione che entrambi gli argomenti di f sono non-negativi ed almeno uno di loro è maggiore di 3 ?

1.

```
int f(in x, int y)
{
assert( (x >= 0) && (y >= 0) && ((x > 3) || (y > 3)) );
.....
}
```

2.

```
int f(in x, int y)
{
assert( (x >= 0) && (y >= 0) && ((x >= 3) || (y >= 3)) );
.....
}
```

3.

```
int f(in x, int y)
{
assert( (x > 0) && (y > 0) && ((x >= 3) || (y > 3)) );
.....
}
```

Risposta: The correct answer is:

```
int f(in x, int y)
{
assert( (x >= 0) && (y >= 0) && ((x > 3) || (y > 3)) );
.....
}
```

39) Un test oracle per un programma P è una funzione booleana che ha come inputs gli inputs ed outputs di P e ritorna true se e solo se il valore di output di P (con i dati inputs) è quello atteso dalle specifiche.
Si consideri la seguente funzione C:

```
int f(int x, int y)  {
int z = x;
while ( (x  <= z)  &&  (z <= y) )  { z = z + 1; }
return (z);
}
```

Siano x, y, gli inputs del programma (f nel nostro caso) e z l'output. Assumendo il programma corretto, quale delle seguenti funzioni booleane F(x, y, z) è un test oracle per la funzione f.

1.

F(x, y, z) = (z == y + 1)

2.

F(x, y, z) = if (x > y) then (z == x) else (z == y + 1)

3.

F(x, y, z) = if (x > y) then (z == x + 1) else (z == y + 1)

Risposta: The correct answer is: F(x, y, z) = if (x > y) then (z == x) else (z == y + 1)

40) Il branch coverage di un insieme di test cases è la percentuale di branch del programma che sono attraversati da almeno un test case.
Si consideri la seguente funzione C:

```
int f(int x, int y)  {
 if (x - y <= 0)   { if (x + y - 1 >= 0)  return (1); else return (2); }
  else {if (2*x + y - 5 >= 0)  return (3); else return (4); }
```

```
} /* f() */
```
Quale dei seguenti test sets consegue una branch coverage del 100% ?

1.
Test set: {x=1, y=1}, {x=2, y=2}, {x=2, y=1}, {x=2, y=0}.
2.
Test set: {x=1, y=1}, {x=0, y=0}, {x=2, y=1}, {x=2, y=3}.
3.
Test set: {x=1, y=1}, {x=0, y=0}, {x=2, y=1}, {x=2, y=0}.
Risposta: The correct answer is: Test set: {x=1, y=1}, {x=0, y=0}, {x=2, y=1}, {x=2, y=0}.

41) Si consideri il seguente requisito:
RQ: Dopo 40 unità di tempo dall'inizio dell'esecuzione vale la seguente proprietà:
se 10 unità di tempo nel passato x era maggiore di 1 allora ora y è nonegativa.
Tenendo presente che, al tempo time, delay(z, w) ritorna 0 se time <= w e ritorna il valore che z aveva al tempo (time - w), se time = w.
Quale dei seguenti monitor meglio descrive il requisito RQ ?

1.
```
class Monitor
InputReal x, y;
OutputBoolean wy;
Boolean wz;
initial equation
wy = false;
equation
wz = (time > 40) and (delay(x, 10) > 1) and (y >= 0);
algorithm
when edge(wz) then
wy := true;
end when;
end Monitor;
```
2.
```
class Monitor
InputReal x, y;
OutputBoolean wy;
Boolean wz;
initial equation
wy = false;
equation
wz = (time > 40) and (delay(x, 10) > 1) and (y < 0);
algorithm
when edge(wz) then
wy := true;
end when;
end Monitor;
```
3.
```
class Monitor
InputReal x, y;
OutputBoolean wy;
Boolean wz;
initial equation
wy = false;
equation
```

```
wz = (time > 40) or (delay(x, 10) > 1) or (y < 0);
algorithm
when edge(wz) then
wy := true;
end when;
end Monitor;
```

Risposta: The correct answer is: class Monitor

```
InputReal x, y;
OutputBoolean wy;
Boolean wz;
initial equation
wy = false;
equation
wz = (time > 40) and (delay(x, 10) > 1) and (y < 0);
algorithm
when edge(wz) then
wy := true;
end when;
end Monitor;
```

42) Una Condition è una proposizione booleana, cioè una espressione con valore booleano che non può essere decomposta in espressioni boolean più semplici. Ad esempio, (x + y <= 3) è una condition.

Una Decision è una espressione booleana composta da conditions e zero o più operatori booleani. Ad esempio, sono decisions:
(x + y <= 3)
((x + y <= 3) || (x - y > 7))
Un insieme di test cases T soddisfa il criterio di Condition/Decision coverage se tutte le seguenti condizioni sono soddisfatte:

1) Ciascun punto di entrata ed uscita nel programma è eseguito in almeno un test;
2) Per ogni decision d nel programma, per ogni condition c in d, esiste un test in T in cui c è true ed un test in T in cui c è false.
3) Per ogni decision d nel programma, esiste un test in T in cui d è true ed un test in T in cui d è false.

Si consideri la seguente funzione:

```
int f(int a, int b, int c)
{    if ( (a - 100 >= 0) && (b - c - 1 <= 0) )
        return (1);    // punto di uscita 1
     else if ((b - c - 1 <= 0)  || (b + c - 5 >= 0)
)
         then return (2);   // punto di uscita 2
         else return (3);   // punto di uscita 3
}
```

Quale dei seguenti test set soddisfa il criterio della Condition/Decision coverage ?

1. (a=200, b = 0, c = 1), (a=50, b = 5, c = 0), (a=50, b = 0, c = 5).
2. (a=200, b = 0, c = 1), (a=50, b = 4, c = 0), (a=200, b = 4, c = 0)
3. (a=200, b = 0, c = 1), (a=50, b = 5, c = 0), (a=50, b = 3, c = 0).

Risposta: The correct answer is: (a=200, b = 0, c = 1), (a=50, b = 5, c = 0), (a=50, b = 3, c = 0).

43) Si consideri il seguente modello Modelica. Quale dei seguenti UML state diagram lo rappresenta correttamente ?

```
block FSA  //  Finite State Automaton
```

```
/* connector declarations outside this block:
connector InputInteger = input Integer;
connector OutputInteger = output Integer;
*/

InputInteger u; // external input
OutputInteger x; // state
parameter Real T = 1;

algorithm

when initial() then
x := 0;

elsewhen sample(0,T) then

if (pre(x) == 0) and (pre(u) == 0) then x := 2;
elseif (pre(x) == 0) and (pre(u) == 1) then x := 4;
elseif (pre(x) == 0) and (pre(u) == 2) then x := 1;
elseif (pre(x) == 1) and (pre(u) == 0) then x := 2;
elseif (pre(x) == 1) and (pre(u) == 1) then x := 4;
elseif (pre(x) == 1) and (pre(u) == 2) then x := 4;
elseif (pre(x) == 2) and (pre(u) == 0) then x := 1;
elseif (pre(x) == 2) and (pre(u) == 1) then x := 1;
elseif (pre(x) == 2) and (pre(u) == 2) then x := 0;
elseif (pre(x) == 3) and (pre(u) == 0) then x := 1;
elseif (pre(x) == 3) and (pre(u) == 2) then x := 0;
elseif (pre(x) == 4) and (pre(u) == 1) then x := 0;
elseif (pre(x) == 4) and (pre(u) == 2) then x := 1;
else x := pre(x); // default
end if;

end when;
end FSA;
```

44) Quale dei seguenti modelli Modelica rappresenta lo state diagram in figura ?

2.

```
block FSA  //  Finite State Automaton

/* connector declarations outside this block:
connector InputInteger = input Integer;
connector OutputInteger = output Integer;
*/

InputInteger u; // external input
OutputInteger x; // state
parameter Real T = 1;

algorithm

when initial() then
```

```
x := 0;

elsewhen sample(0,T) then

if (pre(x) == 0) and (pre(u) == 1) then x := 2;
elseif (pre(x) == 0) and (pre(u) == 2) then x := 4;
elseif (pre(x) == 1) and (pre(u) == 1) then x := 4;
elseif (pre(x) == 2) and (pre(u) == 0) then x := 0;
elseif (pre(x) == 2) and (pre(u) == 1) then x := 4;
elseif (pre(x) == 2) and (pre(u) == 2) then x := 1;
elseif (pre(x) == 3) and (pre(u) == 0) then x := 2;
elseif (pre(x) == 3) and (pre(u) == 1) then x := 0;
elseif (pre(x) == 3) and (pre(u) == 2) then x := 2;
elseif (pre(x) == 4) and (pre(u) == 0) then x := 1;
elseif (pre(x) == 4) and (pre(u) == 1) then x := 2;
elseif (pre(x) == 4) and (pre(u) == 2) then x := 1;
else x := pre(x); // default
end if;

end when;
end FSA;
```

45) La state coverage di un insieme di test cases (cioè sequenze di inputs) per uno state diagram è la percentuale di stati (inclusi START ed END) raggiunti almeno una volta.
Si consideri lo state diagram in figura
Si consideri il seguente insieme di test cases:

Test case 1: act2 act1 act2 act2
Test case 2: act2 act0 act2 act0 act2
Test case 3: act2 act0 act2 act0 act1

Quale delle seguenti è la migliore stima della state coverage per i test cases di cui sopra

1.
State coverage: 87%
2.
State coverage: 60%
3.
State coverage: 100%
Risposta: The correct answer is: State coverage: 87%

46) Pre-condizioni, invarianti e post-condizioni di un programma possono essere definiti usando la macro del C assert() (in <assert.h>). In particolare, assert(expre) non fa nulla se l'espressione expre vale TRUE (cioè non è 0), stampa un messaggio di errore su stderr e abortisce l'esecuzione del programma altrimenti.
Si consideri la funzione C
int f(int x, int y) { ..... }
Quale delle seguenti assert esprime l'invariante che le variabili locali z e w di f() hanno somma minore di 1 oppure maggiore di 7 ?

1.

```
int f(in x, int y)
{
int z, w;
assert( (z + w < 1) || (z + w > 7));
```

```
.....
}
```
2.
```
int f(in x, int y)
{
int z, w;
assert( (z + w <= 1) || (z + w >= 7));
.....
}
```
3.
```
int f(in x, int y)
{
int z, w;
assert( (z + w > 1) || (z + w < 7));
.....
}
```
Risposta: The correct answer is:

```
int f(in x, int y)

{
int z, w;
assert( (z + w < 1) || (z + w > 7));
.....
}
```

47) Quale dei seguenti modelli Modelica rappresenta lo state diagram in figura ?

1.
```
block FSA  //  Finite State Automaton

/* connector declarations outside this block:
connector InputInteger = input Integer;
connector OutputInteger = output Integer;
*/

InputInteger u; // external input
OutputInteger x; // state
parameter Real T = 1;

algorithm

when initial() then
x := 0;

elsewhen sample(0,T) then

if (pre(x) == 0) and (pre(u) == 1) then x := 4;
elseif (pre(x) == 0) and (pre(u) == 2) then x := 4;
elseif (pre(x) == 1) and (pre(u) == 0) then x := 4;
elseif (pre(x) == 1) and (pre(u) == 2) then x := 2;
elseif (pre(x) == 2) and (pre(u) == 0) then x := 4;
elseif (pre(x) == 2) and (pre(u) == 1) then x := 0;
```

```
elseif (pre(x) == 2) and (pre(u) == 2) then x := 3;
elseif (pre(x) == 3) and (pre(u) == 0) then x := 0;
elseif (pre(x) == 3) and (pre(u) == 2) then x := 0;
elseif (pre(x) == 4) and (pre(u) == 2) then x := 1;
else x := pre(x); // default
end if;

end when;
end FSA;
```

2.

```
block FSA  //  Finite State Automaton

/* connector declarations outside this block:
connector InputInteger = input Integer;
connector OutputInteger = output Integer;
*/

InputInteger u; // external input
OutputInteger x; // state
parameter Real T = 1;

algorithm

when initial() then
x := 0;

elsewhen sample(0,T) then

if (pre(x) == 0) and (pre(u) == 0) then x := 3;
elseif (pre(x) == 0) and (pre(u) == 1) then x := 2;
elseif (pre(x) == 0) and (pre(u) == 2) then x := 3;
elseif (pre(x) == 1) and (pre(u) == 0) then x := 4;
elseif (pre(x) == 1) and (pre(u) == 1) then x := 3;
elseif (pre(x) == 1) and (pre(u) == 2) then x := 2;
elseif (pre(x) == 2) and (pre(u) == 0) then x := 0;
elseif (pre(x) == 3) and (pre(u) == 0) then x := 1;
elseif (pre(x) == 3) and (pre(u) == 1) then x := 2;
elseif (pre(x) == 4) and (pre(u) == 1) then x := 0;
else x := pre(x); // default
end if;

end when;
end FSA;
```

48) Si consideri il seguente requisito:
RQ1: Durante l'esecuzione del programma (cioè per tutti gli istanti di tempo positivi) la variabile x è sempre nell'intervallo [0, 5] oppure [10, 15]
Quale dei seguenti monitor meglio descrive il requisito RQ1 ?

1.

```
class Monitor

InputReal x;  // plant output
OutputBoolean y;
```

```
Boolean z;
initial equation
y = false;
equation
z = (time > 0) and ((x >= 0) or (x <= 5))  and  ((x >= 10) or (x <= 15)) );
algorithm
when edge(z) then
y := true;
end when;
end Monitor;
```

2.

```
class Monitor

InputReal x;  // plant output
OutputBoolean y;

Boolean z;
initial equation
y = false;
equation
z = (time > 0) and ((x >= 5) or (x <= 0))  and  ((x >= 15) or (x <= 10)) ;
algorithm
when edge(z) then
y := true;
end when;
end Monitor;
```

3.

```
class Monitor

InputReal x;  // plant output
OutputBoolean y;

Boolean z;
initial equation
y = false;
equation
z = (time > 0) and ( ((x >= 0) and (x <= 5))  or ((x >= 10) and (x <= 15)) );
algorithm
when edge(z) then
y := true;
end when;
end Monitor;
```

Risposta: The correct answer is: class Monitor

```
InputReal x;  // plant output
OutputBoolean y;

Boolean z;
initial equation
y = false;
equation
z = (time > 0) and ((x >= 5) or (x <= 0))  and  ((x >= 15) or (x <= 10)) ;
algorithm
```

```
when edge(z) then
y := true;
end when;
end Monitor;
```

49) Il branch coverage di un insieme di test cases è la percentuale di branch del programma che sono attraversati da almeno un test case.
Si consideri la seguente funzione C:

```
-----------
int f(int x, int y)  {
 if (x - y - 2 <= 0)   { if (x + y - 1 >= 0)  return (1); else return (2); }
  else {if (x + 2*y - 5 >= 0)  return (3); else return (4); }
 }  /* f()  */
```

Si considerino i seguenti test cases: {x=1, y=2}, {x=0, y=0}, {x=5, y=0}, {x=3, y=0}.
Quale delle seguenti è la branch coverage conseguita?

1.
80%
2.
100%
3.
50%

Risposta: The correct answer is: 100%

50) "Ogni giorno, per ciascuna clinica, il sistema genererà una lista dei pazienti che hanno un appuntamento quel giorno."
La frase precedente è un esempio di:

1.
Requisito di performance.
2.
Requisito non-funzionale.
3.
Requisito funzionale.

Risposta: The correct answer is: Requisito funzionale.