CS536 Final Exam Practice Questions - SOLUTIONS

All of the questions and solutions in this document were submitted by students (thank you to those who submitted!). Stefan has checked them over, but not extremely carefully, and there may be errors. If you're unsure if something is an error, feel free to post to Discord.

Short Answer/Multiple Choice/True/False

1. Enumerate the similarities and differences between verification and testing

Answer:

Similarities	**Differences**
May be used to improved confidence in code	Verification can often be more convoluted than testing.
Offers no explicit guarantees	Verification validates requirements, Testing validates that software works correctly.

2. If {p1} s1 {q1} & {p2} s2 {q2} are executed in sequence are valid. Now if p1=p2 & q1=q2 can these statements be combined to run in parallel and will the statement hold. Provide an example.

No this will not hold.
Ex: $\{x>0\}$ x:=x-1 $\{x \geq 0\}$ is valid when run individually.
However, $\{x>0\}$ x:=x-1 || x:=x-1 $\{x \geq 0\}$ can never be valid when run in parallel.
The second x:=x-1 has a precondition $\{x \geq 0\}$ which is too weak.

3. Which of the following is incorrect ?

A. (p ∧ q ∨ r) -> s <-> t ≡ (((p ∧ q) ∨ r) -> s) <-> t
B. p -> (q -> s ∧ t) ≡ p -> (q -> (s ∧ t))
C. (((p ∨ q) ∨ r) ∨ s) ≡ (p ∨ (q ∨ (r ∨ s)))
D. All are correct

Answer - C
Explanation - Parenthesis changes the order of calculation and no longer are syntactically equal.
Eg -
2 + 2 - 3 ≡ (2+2) - 3 !≡ 2 + (2-3)

4. Which of the following gives information about S ?

A⊨[F] S [T]

B⊨[T] S [T]

C⊨[F] S [F]

Answer - B
Explanation - We get to know “S always terminates”

5. To make the below statement always a valid triple, what blank should hold?

   {______} x: = y/z {x>0}

6. A loop invariant must be true
   a. before the loop starts
   b. before each iteration of the loop
   c. after the loop terminates
   d. all of the above

 Answer: d

7. which of the following is incorrect?
   A. p->(q->s /\ t) ≡ p->(q->(s /\ t))
   B. (p \/ q)/\ r ≡ p\/ q /\ r
   C. s -> q <-> t ≡ (s->q) <-> t
   D. p <-> q <=> (p -> q) /\ (q -> p)

   The correct answer is B.

8. for a program S ≡ { y := 4 || y := y -3; y := y * 3}, which of following is not in set M(S, {y = 3})
   A. { y = 4}
   B. {y = 3}
   C. { y = 12}
   D. {y = 8}
   The correct answer is D.

9. Which of the following expressions are syntactically equivalent (≡) ?

a) $a \rightarrow b \rightarrow s \equiv (a \rightarrow b) \rightarrow s$

b) $a \rightarrow b \rightarrow s \equiv a \rightarrow (b \rightarrow s)$

c) $(a \rightarrow b) \rightarrow s \equiv a \rightarrow (b \rightarrow s)$

d) $a \rightarrow b \rightarrow s \equiv ((a \rightarrow b)) \rightarrow (s)$

Solution:
$a \rightarrow b \rightarrow s \equiv a \rightarrow (b \rightarrow s)$

True/False

10. [T] S [T] is valid for any S.
    False: It cannot be true for any S, S can end in divergence or error and hence, triple cannot be satisfied.
11. [x= 2k2] y = sqrt(x) [y>=0] is invalid.
    False, above triple is valid because program will never go in divergence or error. Also p and q are will be true after execution of program
12. If none of the guards of a non deterministic while loop hold true then we raise an runtime error.
    False
13. We should think of non- determinism as random(Hint:With respect to this subject as discussed in class)
    False. We should think of non-determinism as unpredictable.
14. In a branch, If none of the guards are true then it will evaluate to runtime error.
    True
15. Loop bound can be a constant.
    False
16. Loop invariant is a condition which remains true before and after loop execution.
    True
17. To disprove a conjecture, it is necessary to prove that it is false in all cases.

False : To prove a conjecture, it is necessary to prove that it is true in all cases;
to disprove a conjecture, it is sufficient to find a single case where it is false

18. ~(E i.j ≤ i < k.a[i]=0) is equivalent to (V i.j ≤ i < k.a[i]=0)

False : ~(E i.j ≤ i < k.a[i]=0) is equivalent to V i.j ≤ i < k.a[i]!=0

19. if wlp(flip, head V tail) = T then we can say wlp(flip,head) V wlp(flip,tail) = T

False : It is guaranteed that flipping the coin will give head or a tail ,therefore
wlp(flip, head v tail) = T
There is no guarantee that flipping the coin will only give head, therefore
wlp(flip,head) = F.
& There is no guarantee that flipping the coin will only give tail, therefore
wlp(flip,tail) = F.
Hence wlp(flip,head) V wlp(flip,tail) = F V F = F

20. Programs always terminate in some state.

False, they could terminate in ⊥d or ⊥e, which are not states

21. Using await instead of wait or a busy while loop removes the possibility of deadlock.

False, the condition that is being waited on may never become true, which will cause deadlock

22. The following two proof outlines are interference-free:

$\{x = 0 \wedge y = 0\}$ $x := x + 1$ $\{x = 1 \wedge y = 0\}$ $y := y + 1$ $\{x = 1 \wedge y = 1\}$
$\{y = 0)$ if $y \geq 0$ then $\{z := -1\}$ else $\{z := 1\}$ $\{z = -1\}$
False. There is interference between $\{x = 1 \wedge y = 0\}$ $y := y + 1$ in proof outline 1 and $y = 0$ in proof outline 2 because y will no longer be equal to 0 if the first thread sets y to 1 before the condition is evaluated

23. In a non-deterministic program, if five different guards are true, each branch as a 20% chance of being executed.

False. In a non-deterministic program, if several guards are true, select non-deterministically one of the corresponding guarded statements and execute it.

24. [F] In a non-deterministic program, if several guards are true, we must always run the first branch.
In a non-deterministic program, if several guards are true, select non-deterministically one of the corresponding guarded statements and execute it.

25. If $\sigma \models \{p\}\ S\ \{q\}$, then $\sigma \models p$.
False; $\sigma \models \{p\}\ S\ \{q\}$ does not imply $\sigma \models p$. (It doesn't imply $\sigma \not\models p$ either.)

26. If $\sigma \models \{p\}\ S\ \{q\}$ and $\sigma \models p$, then every state in $M(S, \sigma)$ either $\in \{\bot d , \bot e\}$ or satisfies q
True; if $\{p\}\ S\ \{q\}$ is partially correct and we run S in a state satisfying p, then either S causes an error or terminates in a state satisfying q.

27. A Hoare triple is satisfied if it has a contradiction as a precondition.

True. Having a precondition as false will result in a satisfied Hoare triple. A contradiction is always false.

28. A loop invariant statement proves that a while loop will terminate.

A) False. You need to have a decrementing function to prove the loop terminates

29. $\{x=3, x=5\}$ is a valid state for a program.
A) False. This would be an ill-formed state. There can only be one value per variable in the state.

Proofs and Proof outlines

30. Convert the following minimal proof outline to a full proof outline by adding the weakest preconditions and/or strongest postconditions. Proof obligations should appear in the proof outline as weakenings ($p \Rightarrow q$), but you do not need to prove them explicitly.

Given function fact(a) computes the factorial of a. The program below returns the binomial expansion of (1+x) where n is non-negative.

Binomial expansion formula: $(1+x)^n \; = \; 1 + nx + \frac{n(n-1)}{2!}x^2 + \frac{n(n-1)(n-2)}{3!}x^3 + \ldots$.

```
                                            $\{n \geq 0\}$
        $i := \underline{0}$
        $s := \underline{1}$
        $k := \underline{n}$
$\{\mathbf{inv}\ \boldsymbol{i \leq n \wedge s = (1+x)^i \wedge k \geq n}\}$
    $while(i < n)\ \{$
        $s := s + \frac{k}{fact(i+1)} x^{i+1};$
        $i := i + 1$
        $k := k(k - i)$
    $\}$
                                            $\{s = \ (1+x)^n\}$
```

“Solution”

```
                                            $\{n \geq 0\}$
        $i := \underline{0}$                $\{n \geq 0 \wedge i = 0\}$
        $s := \underline{1}$                $\{n \geq 0 \wedge i = 0 \wedge s = 1\}$
        $k := \underline{n}$                $\{n \geq 0 \wedge i = 0 \wedge s = 1 \wedge k = n\}$
                                            $\Rightarrow \{i \leq n \wedge s = (1+x)^i \wedge k \geq n\}$
$\{\mathbf{inv}\ \boldsymbol{i \leq n \wedge s = (1+x)^i \wedge k \geq n}\}$
```

$while(i < n)\ \{$ $\quad\{i \leq n \wedge s = (1+x)^i \wedge k \geq n \wedge i < n\}$

$$\Rightarrow \{i \leq n \wedge i + 1 < n \wedge s + \frac{k}{fact(i+1)} x^{i+1} = (1+x)^{i+1} \wedge k(k - i + 1) \geq n\}$$

$s := s + \frac{k}{fact(i+1)} x^{i+1};$ $\quad\{i \leq n \wedge i + 1 < n \wedge s = (1+x)^{i+1} \wedge k(k - i + 1) \geq n\}$

$i := i + 1$ $\quad\{i \leq n \wedge s = (1+x)^i \wedge k(k - i) \geq n\}$

$k := k(k - i)$ $\quad\{i \leq n \wedge s = (1+x)^i \wedge k \geq n\}$

$\}$ $\quad\{i = n \wedge s = (1+x)^i \wedge k \geq n\} \Rightarrow \{s = (1+x)^n\}$

$$\{s = (1+x)^n\}$$

31. Findout loop invariant and loop bounds

```
requires N >= 0
ensures t == N * (N + 1) / 2
{
t := 0;
var n := 0;
while n < N
invariant ?
decreases ?
{
n,
t := n + 1,
t + n + 1;
}
}
Ans: invariant 0 <= n <= N && t == n * (n + 1) / 2
decreases N-n
```

32. a. Fill in the following .dfy program to make a totally correct method indicating whether or not x is in an array a

```
method x_in_array(x: int, a:array<int>) returns (b:bool)
ensures b ==>____________________________
{
   var iter := 0;
   b := false;
   while ____________________
   decreases ________________
   invariant ((iter <= a.Length) && (b ==> ______________________))
   {
      b := a[iter] == x;
      ___________________;
```

```
    }
}
```

Sol:

```
method x_in_array(x: int, a:array<int>) returns (b:bool)
ensures b ==> exists i :: 0 <= i < a.Length && x == a[i]
{
    var iter := 0;
    b := false;
    while iter < a.Length
    decreases a.Length - iter
    invariant ((iter <= a.Length) && (b ==> exists i :: 0 <= i < iter && x == a[i]))
    {
        b := a[iter] == x;
        iter := iter + 1;
    }
}
```

b. Now, write an *ensures* clause which sets a boolean iff x is in an array a *twice*

b ==> ____

Sol:

```
b ==> (exists i,j :: 0 <= i < a.Length && 0 <= j < a.Length && x == a[j] && x == a[i] && i != j)
```

c. Now, fill in the following .dfy program to make a totally correct method indicating whether or not x is in an array a *twice*

Note the invariant for this while loop will be long and won't fit into the given spaces

```
method x_in_array_2_times(x: int, a:array<int>) returns (b:bool)
ensures b ==> ______________________________________________________________________
{
    var iter := 0;
    var seen_once := false;
    b := false;
    while _________________
    decreases ___________________
    invariant (iter <= a.Length && ________________________________________________
    && b ==> _________________________________________________________)
    {
        if (a[iter] == x && seen_once) {b := true;}
        if (a[iter] == x && !seen_once) {seen_once := true;}

        iter := iter + 1;
    }
}
```

Sol:

```
method x_in_array_2_times(x: int, a:array<int>) returns (b:bool)
ensures b ==> (exists i,j :: 0 <= i < a.Length && 0 <= j < a.Length && x == a[j] && x == a[i] && i != j)
{
    var iter := 0;
    var once := false;
    b := false;
    while iter < a.Length
    decreases a.Length - iter
    invariant iter <= a.Length
    invariant once ==> exists i :: 0 <= i < iter && a[i] == x
    invariant b ==> (exists i,j :: 0 <= i < iter && 0 <= j < iter && x == a[j] && x == a[i] && i != j)
    {
        if (a[iter] == x && once) {b := true;}
        if (a[iter] == x && !once) {once := true;}

        iter := iter + 1;
    }
}
```

d. Alternatively, instead of working on c as a whole, we can just use a buggy version of the same program. The following is an incorrect version of a program which asks "is x in this array at least twice?"
Which invariant or ensures statement does not hold?
Using words, why does this invariant or ensures statement not hold, and how do we fix the code so it DOES hold?

```
method x_in_array_2_times(x: int, a:array<int>) returns (b:bool)
ensures b ==> (exists i,j :: 0 <= i < a.Length && 0 <= j < a.Length && x == a[j] && x == a[i] && i != j)
{
    var iter := 0;
    var once := false;
    b := false;
    while iter < a.Length
    decreases a.Length - iter
    invariant iter <= a.Length
    invariant once ==> exists i :: 0 <= i < iter && a[i] == x
    invariant b ==> (exists i,j :: 0 <= i < iter && 0 <= j < iter && x == a[j] && x == a[i] && i != j)
    {
        if (a[iter] == x && !once) {once := true;}
        if (a[iter] == x && once) {b := true;}

        iter := iter + 1;
    }
}
```

Sol:
The while loop's invariant b ==> (exists i,j :: 0 <= i < iter && 0 <= j < iter && x == a[j] && x == a[i] && i != j)
does not hold

The issue is the order of the two if-statements inside the while loop. They are in the wrong order, and b will be set when we only encounter x once as a result. We either need:

```
    if (a[iter] == x && once) {b := true;}
    if (a[iter] == x && !once) {once := true;}
```

Or, we need to use if … and else if …

33.

a) Give appropriate loop invariants for any loops.

b) Write a full proof for the program. You may write a full proof outline or a Hilbert-style proof.

This program computes the sum $1 + 2_1 + ... + 2_{n-1}$.

Precondition $\{n \geq 0\}$

$i := 0;$

$s := 0;$

$k := 1;$

Program while $(i < n)$ {

$s := s + k;$

$k := k * 2;$

$i := i + 1$

}

Postcondition $\{s = 2_n - 1\}$

Solution.

$\{n \geq 0\}$

$i := 0; \{n \geq 0 \wedge i = 0\}$

$s := 0; \{n \geq 0 \wedge i = 0 \wedge s = 0\}$

$k := 1; \{n \geq 0 \wedge i = 0 \wedge s = 0 \wedge k = 1\}$

$\{inv\ i \leq n \wedge k = 2_i \wedge s = 2_i - 1\}$

$\{dec\ n - i\}$

while $(i < n)$ { $\{i < n \wedge i \leq n \wedge k = 2_i \wedge s = 2_i - 1\} \Rightarrow \{i < n \wedge k = 2_i \wedge s = 2_i - 1\}$

$s := s + k; \{i < n \wedge k = 2_i \wedge s_0 = 2_i - 1 \wedge s = s_0 + k\} \Rightarrow \{i < n \wedge k = 2_i \wedge s = 2_{i+1} - 1\}$

$k := k * 2; \{i < n \wedge k_0 = 2_i \wedge k = k_0 * 2 \wedge s = 2_{i+1} - 1\}$

$\Rightarrow \{i + 1 < n + 1 \wedge k = 2_{i+1} \wedge s = 2_{i+1} - 1\}$

$i := i + 1\ \{i < n + 1 \wedge k = 2_i \wedge s = 2_i - 1\} \Rightarrow \{i \leq n \wedge k = 2_i \wedge s = 2_i - 1\}$

} $\{i \geq n \wedge i \leq n \wedge k = 2_i \wedge s = 2_i - 1\} \Rightarrow \{s = 2_n - 1\}$

34.

Binary Search

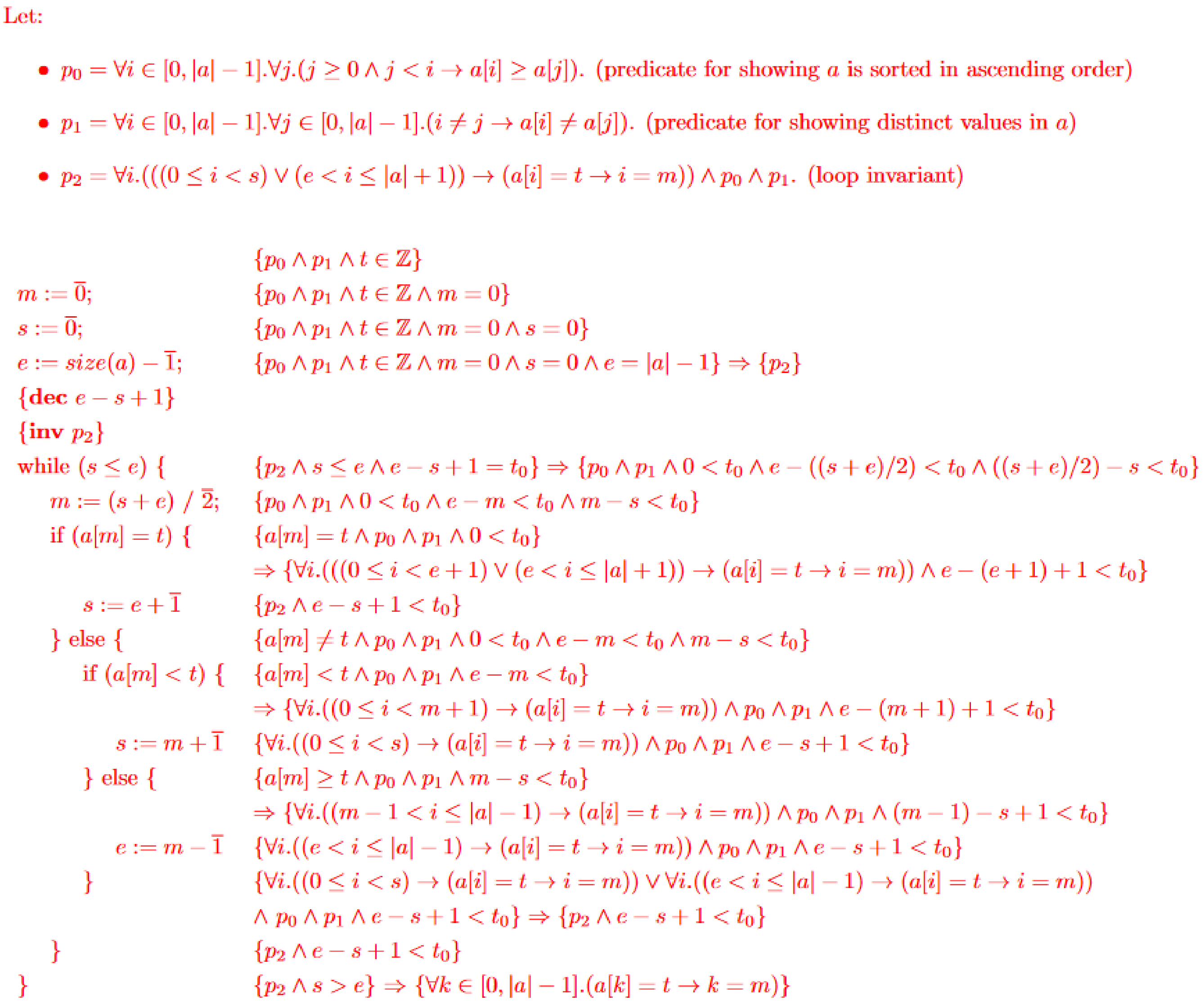
The following program performs a binary search on the array a for an integer value t. a is not guaranteed to have the value t. Assume the values in a are **distinct**. Write a **full** proof outline for the program. Show in the precondition that a is sorted in ascending order. Include an appropriate loop invariant and loop bound.

Precondition: { ______ $\wedge \forall i \in [0, |a| - 1].\forall j \in [0, |a| - 1].(i \neq j \rightarrow a[i] \neq a[j]) \wedge t \in \mathbb{Z}$ }

Program:

```
$m := \overline{0};$
$s := \overline{0};$
$e := size(a) - \overline{1};$
{dec ________}
{inv ________}
while $(s \leq e)$ {
    $m := (s + e)\ /\ \overline{2};$
    if $(a[m] = t)$ {
        $s := e + \overline{1}$
    } else {
        if $(a[m] < t)$ {
            $s := m + \overline{1}$
        } else {
            $e := m - \overline{1}$
        }
    }
}
```

Postcondition: $\{\forall k \in [0, |a| - 1].(a[k] = t \rightarrow k = m)\}$



Let:

- $p_0 = \forall i \in [0, |a|-1].\forall j.(j \geq 0 \wedge j < i \rightarrow a[i] \geq a[j])$. (predicate for showing a is sorted in ascending order)
- $p_1 = \forall i \in [0, |a|-1].\forall j \in [0, |a|-1].(i \neq j \rightarrow a[i] \neq a[j])$. (predicate for showing distinct values in a)
- $p_2 = \forall i.(((0 \leq i < s) \vee (e < i \leq |a|+1)) \rightarrow (a[i] = t \rightarrow i = m)) \wedge p_0 \wedge p_1$. (loop invariant)

```
                            ${p_0 \wedge p_1 \wedge t \in \mathbb{Z}}$
$m := \overline{0}$;                  ${p_0 \wedge p_1 \wedge t \in \mathbb{Z} \wedge m = 0}$
$s := \overline{0}$;                  ${p_0 \wedge p_1 \wedge t \in \mathbb{Z} \wedge m = 0 \wedge s = 0}$
$e := size(a) - \overline{1}$;        ${p_0 \wedge p_1 \wedge t \in \mathbb{Z} \wedge m = 0 \wedge s = 0 \wedge e = |a|-1} \Rightarrow {p_2}$
${\textbf{dec } e - s + 1}$
${\textbf{inv } p_2}$
while ($s \leq e$) {        ${p_2 \wedge s \leq e \wedge e - s + 1 = t_0} \Rightarrow {p_0 \wedge p_1 \wedge 0 < t_0 \wedge e - ((s+e)/2) < t_0 \wedge ((s+e)/2) - s < t_0}$
   $m := (s + e) / \overline{2}$;    ${p_0 \wedge p_1 \wedge 0 < t_0 \wedge e - m < t_0 \wedge m - s < t_0}$
   if ($a[m] = t$) {         ${a[m] = t \wedge p_0 \wedge p_1 \wedge 0 < t_0}$
                            $\Rightarrow {\forall i.(((0 \leq i < e+1) \vee (e < i \leq |a|+1)) \rightarrow (a[i] = t \rightarrow i = m)) \wedge e - (e+1) + 1 < t_0}$
      $s := e + \overline{1}$          ${p_2 \wedge e - s + 1 < t_0}$
   } else {                 ${a[m] \neq t \wedge p_0 \wedge p_1 \wedge 0 < t_0 \wedge e - m < t_0 \wedge m - s < t_0}$
      if ($a[m] < t$) {      ${a[m] < t \wedge p_0 \wedge p_1 \wedge e - m < t_0}$
                            $\Rightarrow {\forall i.((0 \leq i < m+1) \rightarrow (a[i] = t \rightarrow i = m)) \wedge p_0 \wedge p_1 \wedge e - (m+1) + 1 < t_0}$
         $s := m + \overline{1}$       ${\forall i.((0 \leq i < s) \rightarrow (a[i] = t \rightarrow i = m)) \wedge p_0 \wedge p_1 \wedge e - s + 1 < t_0}$
      } else {              ${a[m] \geq t \wedge p_0 \wedge p_1 \wedge m - s < t_0}$
                            $\Rightarrow {\forall i.((m - 1 < i \leq |a|-1) \rightarrow (a[i] = t \rightarrow i = m)) \wedge p_0 \wedge p_1 \wedge (m-1) - s + 1 < t_0}$
         $e := m - \overline{1}$       ${\forall i.((e < i \leq |a|-1) \rightarrow (a[i] = t \rightarrow i = m)) \wedge p_0 \wedge p_1 \wedge e - s + 1 < t_0}$
      }                     ${\forall i.((0 \leq i < s) \rightarrow (a[i] = t \rightarrow i = m)) \vee \forall i.((e < i \leq |a|-1) \rightarrow (a[i] = t \rightarrow i = m))$
                            $\wedge p_0 \wedge p_1 \wedge e - s + 1 < t_0} \Rightarrow {p_2 \wedge e - s + 1 < t_0}$
   }                        ${p_2 \wedge e - s + 1 < t_0}$
}                           ${p_2 \wedge s > e} \Rightarrow {\forall k \in [0, |a|-1].(a[k] = t \rightarrow k = m)}$
```

35.Factorial:(with Loop invariant, loop bound and full proof outline)

Pre condition: {N ≥ 1}   { 1 = 1! Ù 0 ≤ 1 ≤ N }

i := 1   { 1 = i! Ù 0 ≤ i ≤ N }

f := 1   { f = i! Ù 0 ≤ i ≤ N }

{inv: f = i! Ù 0 ≤ i ≤ N }

{dec: N - i }

while (i < N) {   { f = i! Ù 0 ≤ i ≤ N Ù i < N Ù N - (i + 1) = t } ⇒

$\{f * (i + 1) = (i + 1)!\ \text{Ù}\ 0 \le i + 1 \le N\ \text{Ù}\ i + 1 < N\ \ \text{Ù}\ N - (i + 1) < t_0\}$

i := i + 1 $\{ f * i\ = i\ !\ \text{Ù}\ 0 \le i\ \le N\ \ \text{Ù}\ N - i < \ t_0 \}$

f := f * i $\{ f\ = i\ !\ \text{Ù}\ 0 \le i\ \le N\ \text{Ù}\ N - i < \ t_0 \}$

} $\{ f\ = i\ !\ \text{Ù}\ 0 \le i\ \le N\ \text{Ù}\ i \ge N \}$

Post condition: $\{ f\ = i\ !\ \text{Ù}\ 0 \le i\ \le N\ \text{Ù}\ i \ge N \} \Rightarrow \{ f = N! \}$

36. Find the Loop Invariant and Loop Bound for the outer while loop of the below shown code snippet. The program takes an input value N and returns an array with all the prime factors Assume that next_prime method returns the next possible prime number

```
def factorise(n) {
   prime = 2
   factors = []
   while n > 1{

      while n % prime == 0{
         factors.append(prime)
         n /= prime
      }
      prime = next_prime(prime)
}
   return factors
}
```

- Invariant **product(factors) * n == initial n** i.e product of all the prime factors multiplied by current value of n.
- Bound **decreases n**

Wp/Wlp/Sp/M/Substitution Calculations

37. Calculate wp(if e then {x := (x*10)/10} else {x := (x+10)/10}; x = 5)

≡ wp(if e then {x := (x*10)/10} else {x := (x+10)/10}, wp(x = 5))
≡ wp(if e then {x := (x*10)/10} else {x := (x+10)/10}, x = 5)
≡ (e ⟶ wp(x := (x*10)/10, x = 5)) ∧ (¬e ⟶ wp(x := (x+10)/10, x = 5))
≡ (e ⟶ (x*10)/10 = 5) ∧ (¬e ⟶ (x+10)/10 = 5)

38. sp(m ≤ n ∧ m/n < k, m := m*n; n := m*n)

sp(m > n ∧ m/n < k, m := m*n; n := m*n)
≡ sp(sp(m > n ∧ m/n < k, m := m*n), n := m*n)
≡ sp(m_o > n ∧ m_o/n < k ∧ m = m_o*n, n := m*n)
≡ m_o > n_o ∧ m_o/n_o < k ∧ m = m_o* n_o ∧ n =m* n_o

39.

$$sp(0 < x < 2y + x^2, x \coloneqq x * x; x \coloneqq x + 5)$$

Difficult points and spots:

1. It is a $sp(p, S_1; S_2)$, which requires the formula: $sp(p, S_1; S_2) = sp(sp(p, S_1), S_2)$.
2. While S_1 have changed x through $[x_0/x]$, if S_2 should change the x through $[x_1/x]$.

Answer:

$$\begin{gathered}
sp(0 < x < 2y + x^2, x \coloneqq x * x; x \coloneqq x + 5) \\
= sp(sp(0 < x < 2y + x^2, x \coloneqq x * x), x \coloneqq x + 5) \\
= sp(([x_0/x](0 < x < 2y + x^2) \wedge x = [x_0/x]x * x), x \coloneqq x + 5) \\
= sp\big((0 < x_0 < 2y + {x_0}^2) \wedge (x = {x_0}^2), x \coloneqq x + 5\big) \\
= (0 < x_0 < 2y + {x_0}^2) \wedge [x_1/x](x = {x_0}^2) \wedge x = [x_1/x](x + 5) \\
= 0 < x_0 < 2y + {x_0}^2 \wedge x_1 = {x_0}^2 \wedge x = x_1 + 5
\end{gathered}$$

40.

$$wp\big(if\ x > y, then\{z{:}= y * \sqrt{x-1}\}\ else\{z{:}= 5/x\}, \forall z. x \neq 4 * z\big)$$

Difficult points and spots:

1. $wlp(if\ e\ then\ S_1\ else\ S_2, q)$ should transfer into $(\ e \longrightarrow wlp(S_1, q)) \wedge (\ \neg e \longrightarrow wlp(S_2, q))$
2. $wp(S, q)$ should be transferred into $wlp(S, q) \wedge D(S)$
3. While encountering the $\forall z. f(z)$, we should not change z.

Answer:

$$\begin{gathered}
wp\big(if\ x > y, then\{z{:}= y * \sqrt{x-1}\}\ else\{z{:}= 5/x\}, \forall z. x \neq 4 * z\big) \\
= wlp\big(if\ x > y, then\{z{:}= y * \sqrt{x-1}\}\ else\{z{:}= 5/x\}, \forall z. x \neq 4 * z\big) \wedge D\big(if\ x > y, then\{z{:}= y * \sqrt{x-1}\}\ else\{z{:}= 5/x\}\big) \\
= \big(\ x > y \longrightarrow wlp(z \coloneqq y * \sqrt{x-1}, \forall z. x \neq 4 * z)\big) \wedge (x \leq y \longrightarrow wlp(z \coloneqq 5/x, \forall z. x \neq 4 * z)) \wedge D(x > y) \\
\wedge \big(x > y \longrightarrow D(z \coloneqq y * \sqrt{x-1})\big) \wedge (x \leq y \longrightarrow D(z{:}= 5/x)) \\
= (x > y \longrightarrow \forall z. x \neq 4 * z) \wedge (x \leq y \longrightarrow \forall z. x \neq 4 * z) \wedge (x > y \longrightarrow x \geq 1) \wedge (x \leq y \longrightarrow x \neq 0)
\end{gathered}$$

41. wp(y = x^2; z=sqrt(y), ∀z.|z| = x /\ z <= y)

wlp(y = x^2; z=sqrt(y), ∀z.|z| = x /\ z <= y) /\ D(y = x^2; z=sqrt(y))
wlp(y = x^2, wlp(z=sqrt(y), ∀z.|z| = x /\ z <= y)) /\ D(y = x^2) /\ D(z=sqrt(y))
wlp(y = x^2, ∀z.|z| = x /\ z <= y) /\ T /\ y >= 0 /\ D(y)
(∀z.|z| = x /\ z <= x^2) /\ y >= 0 /\ T
(∀z.|z| = x /\ z <= x^2) /\ y >= 0

42. sp(x=y, x = y * x; y = x * y)

sp(sp(x=y, x = y * x),y = x * y)
sp(x_0 = y /\ x = y * x_0, y = x * y)
x_0 = y_0 /\ x = y_0 * x_0 /\ y = x * y_0

43. wp(n := x/sqrt(k), n > 0)
Ans: wp(n := x/sqrt(k), n > 0)
= wlp(n := x/sqrt(k), n > 0) ∧ D(n := x/sqrt(k))
= [(x/sqrt(k))/n]n > 0 ∧ D(x/sqrt(k))
= (x/sqrt(k)) > 0 ∧ D(x) ∧ D(sqrt(k)) ∧ sqrt(k) ≠ 0
= (x/sqrt(k)) > 0 ∧ T ∧ D(k) ∧ k ≥ 0 ∧ sqrt(k) ≠ 0
= (x/sqrt(k)) > 0 ∧ D(k) ∧ k ≥ 0 ∧ sqrt(k) ≠ 0
= (x/sqrt(k)) > 0 ∧ k ≥ 0 ∧ sqrt(k) ≠ 0

44. wlp($i = sqrt(z); x = z * i^2, x > 0$)

= wlp($i = sqrt(z); wlp(x = z * i^2, x > 0)$)
= wlp($i = sqrt(z); z * i^2 > 0$)
= [sqrt(z) / i] ($z * i^2 > 0$)
= $(z * sqrt(z)^2 > 0) = z^2 > 0$

45. [x+y/z] $\forall x. \exists y. y = \frac{x}{z}$

= $\forall x. \exists y. \left[\frac{i}{x}\right]\left[\frac{j}{y}\right] y = \frac{x}{z}$
= [$(\mathrm{x} + \mathrm{y})/\mathrm{z}]\forall i. \exists j. j = \frac{i}{z}$
= $\forall i. \exists j. j = \frac{i}{x+y}$

46.

Determine the strongest post condition. Assume mod(x) returns the remainder of integer division.
SP(x = 2 , x:=x+1; if mod(x)=0 then {x:=x+1}else{x:=x+2})
A:
=SP(x = 2 , x:=x+1; if mod(x)=0 then {x:=x+1}else{x:=x+2})
=SP(SP(x = 2 , x:=x+1), if mod(x)=0 then {x:=x+1}else{x:=x+2})
=SP((x = 3 ∧ x:=x0+1), if mod(x)=0 then {x:=x+1}else{x:=x+2})
=SP(x = 3 ∧ x:=x0+1 ∧ mod(x)=0,x:=x+1 ∨ SP(x = 3 ∧ x:=x0+1 ∧ mod(x)=0, {x:=x+2})
=SP(x = 3 ∧ x:=x0+1 ∧ mod(x)=0,x:=x+1 ∨ SP(x = 3 ∧ x:=x0+1 ∧ mod(x) != 0, {x:=x+2})

=SP(x = 3 ∧ x:=x0+1 ∧ mod(x) != 0, {x:=x+2})
=x1 = 3 ∧ x1:=x0+1 ∧ mod(x1) != 0 ∧ x:=x1+2

47.

Q:Determine the weakest pre condition.
WP(x:=x-1;x:=1/x,x<1)
A:
WP(x:=x-1;x:=1/x,x<1)
=WLP(x:=x-1;x:=1/x,x<1) ∧ D(x:=x-1;x:=1/x)
=WLP(x:=x-1, WLP(x:=1/x),x<1) ∧ D(x:=x-1) ∧ WP(x:=x-1,D(x:=1/x))
=WLP(x:=x-1, 1/x <1) ∧ D(x-1) ∧ WP(x:=x-1,D(1) ∧ D(x) ∧ x != 0)
=(1/(x-1)) <1) ∧ T ∧ WP(x:=x-1,T ∧ T ∧ x != 0)
=(1/(x-1)) <1) ∧ WP(x:=x-1, x != 0)
=(1/(x-1)) <1) ∧ (x-1 != 0)

48. wlp(i := 2; a[j] := a[i] ∗ i, a[i] = 3)

= wlp(i := 2, wlp(a[j] := a[i] ∗ i, a[i] = 3))
= wlp(i := 2, i = j?a[i] ∗ i = 3 : a[i] = 3)
= j = 2 ? a[2] ∗ 2 = 3 : a[2] = 3
= j = 2 ? a[2] = 3
2 : a[2] = 3

49. wlp(x := y, z := x/y, z ≥ 9)

= wlp(x := y, wlp(z := x
y , z ≥ 9))
= wlp(x := y, x
y ≥ 9)
= y
y ≥ 9
= 1 ≥ 9
= F

50. wp([x > 0 : x = y : x = -y], y = x/y)

Let s = x > 0: x = y : x = -y and q = x/y
wp(s,q) =wlp(s,q) /\ D(s)
wlp(s,q) = wlp([x > 0: x = y : x = -y], y = x/y)
= (x > 0 -> wlp(x=y, y=x/y)) /\ (x ≤ 0 -> wlp(x= -y, y = x/y))
= (x > 0 -> y =1) /\ (x ≤ 0 -> y = -1)
D(s) = D([x > 0 : x = y : x = -y])
= D(x>0) /\ (x > 0 -> D (x=y)) /\ (x ≤ 0 -> D (x=-y))
= T /\ (x>0 -> T) /\ (x ≤ 0 -> T)
= T /\ T /\ T => T
wp(s,q) =wlp(s,q) /\ D(s)
= (x > 0 -> y =1) /\ (x ≤ 0 -> y = -1)

51. wp (if x < 5 then y:= x*x; else y := x+1/z; y >= 9)

Ans:

$\equiv (x < 5 \rightarrow wp\ (y:= x*x, y >= 9)) \wedge (x >= 5 \rightarrow wp\ (y:= x+1/z, y >= 9))$

$\equiv (x < 5 \rightarrow wp\ (y:= x*x, y >= 9)) \wedge (x >= 5 \rightarrow wlp\ (y:= x+1/z, y >= 9) \wedge D(y:= x+1/z))$

$\equiv (x < 5 \rightarrow wp\ (y:= x*x, y >= 9)) \wedge (x >= 5 \rightarrow wlp\ (y:= x+1/z, y >= 9) \wedge D(y:= x+1) \wedge D(z) \wedge z \neq 0)$

$\equiv (x < 5 \rightarrow [x*x/y]\ y >= 9) \wedge (x >= 5 \rightarrow [x+1/y]\ y >= 9 \wedge z \neq 0)$

$\equiv (x < 5 \rightarrow x*x >= 9) \wedge (x >= 5 \rightarrow x+1 >= 9 \wedge z \neq 0)$

52. let σ = {x=5,y=[2;4;5;8],z=1}

A) what is σ[x|->3] [z|->2] [b|->3] ?
B) What is σ[z|->5] (x) ?

Solution):-

A)
= {x=5,y=[2;4;5;8],z=1}[x|->3] [z|->2] [b|->3]

={x=3,y=[2;4;5;8],z=2,b=3}.

B)
=σ[z|->5] (x)
=σ[z|->5] (x) = σ(x) = 5.

53. wp(if x>y then x:=4 else y:=x, y≠x)

Solution:
Using the rule $wp(s, q) = wlp(s, q) \wedge D(s)$,
$wlp(\text{if } x>y \text{ then } x:=4 \text{ else } y:=x, y\neq x) \wedge D(\text{if } x>y \text{ then } x:=4 \text{ else } y:=x)$

Using the rule
$wlp(\text{if } e \text{ then } \{s1\} \text{ else } \{s2\}, q) = (e \rightarrow wlp(s1, q)) \wedge (\neg e \rightarrow wlp(s2, q))$,
$(x>y \rightarrow wlp(x:=4, y\neq x)) \wedge (x\leq y \rightarrow wlp(y:=x, y\neq x)) \wedge D(\text{if } x>y \text{ then } x:=4 \text{ else } y:=x)$

Using the rule $wlp(x := e, q) = [e/x]q$,
$(x>y \rightarrow [4 / x](y\neq x)) \wedge (x\leq y \rightarrow [x / y](y\neq x)) \wedge D(\text{if } x>y \text{ then } x:=4 \text{ else } y:=x)$

Using the rule $[e / x](e1 \text{ op } e2) \equiv [e / x](e1) \text{ op } [e / x](e2)$,

$(x>y \rightarrow y\neq 4) \land (x\leq y \rightarrow x\neq x) \land D(\text{if } x>y \text{ then } x:=4 \text{ else } y:=x)$

Using the rule $D(\text{if } e \text{ then } \{s1\} \text{ else } \{s2\}) = D(e) \land (e \rightarrow D(s1)) \land (\neg e \rightarrow D(s2))$,
$(x>y \rightarrow y\neq 4) \land (x\leq y \rightarrow x\neq x) \land D(x>y) \land (x>y \rightarrow D(x:=4)) \land (x\leq y \rightarrow D(y:=x))$

Using the rule $D(e1 \oplus e2) = D(e1) \land D(e2)$,
$(x>y \rightarrow y\neq 4) \land (x\leq y \rightarrow x\neq x) \land D(x) \land D(y) \land (x>y \rightarrow D(x:=4)) \land (x\leq y \rightarrow D(y:=x))$

Using the rule $D(x := e) = D(e)$,
$(x>y \rightarrow y\neq 4) \land (x\leq y \rightarrow x\neq x) \land D(x) \land D(y) \land (x>y \rightarrow D(4)) \land (x\leq y \rightarrow D(x))$

Using the rule $D(c) = T$,
$(x>y \rightarrow y\neq 4) \land (x\leq y \rightarrow x\neq x) \land D(x) \land D(y) \land (x>y \rightarrow T) \land (x\leq y \rightarrow D(x))$

Using the rule $D(x) = T$,
$(x>y \rightarrow y\neq 4) \land (x\leq y \rightarrow x\neq x) \land T \land T \land (x>y \rightarrow T) \land (x\leq y \rightarrow T)$

Simplifying,
$(x>y \rightarrow y\neq 4) \land (x\leq y \rightarrow x\neq x) \land T \land T \land (x>y \rightarrow T) \land (x\leq y \rightarrow T) \equiv$
$(x\leq y \lor y\neq 4) \land (x>y \lor x\neq x) \land (x\leq y \lor T) \land (x>y \lor T) \equiv$
$(x\leq y \lor y\neq 4) \land (x>y \lor F) \land (T) \land (T) \equiv$
$(x\leq y \lor y\neq 4) \land x>y$

54. sp(y=0, x:=4;if x>y then y:=x else skip)

Solution:

Using the rule sp(p, s1; s2) = sp(sp(p, s1), s2),
sp(sp(y=0, x:=4), if x>y then y:=x else skip)

Using the rule sp(p, x := e) = [x0/x]p ∧ x = [x0/x]e,
sp([x0 / x](y=0) ∧ x = [x0 / x]4, if x>y then y:=x else skip)

Using the substitution rule [e / x](e1 op e2) = [e / x]e1 op [e / x] e2,
sp([x0 / x]y = [x0 / x]0 ∧ x = [x0 / x]4, if x>y then y:=x else skip)

Using the substitution rule [e / x]c = c,
sp([x0 / x]y = 0 ∧ x=4, if x>y then y:=x else skip)

Using the substitution rule [e / x]y = y,
sp(y=0 ∧ x=4, if x>y then y:=x else skip)

Using the rule sp(p, if e then {s1} else {s2}) = sp(p ∧ e, s1) ∨ sp(p ∧ ¬e, s2),
sp((y=0 ∧ x=4) ∧ x>y, y:=x) ∨ sp((y=0 ∧ x=4) ∧ x≤y, skip)

Using the rule sp(p,skip) = p,
sp((y=0 ∧ x=4) ∧ x>y, y:=x) ∨ ((y=0 ∧ x=4) ∧ x≤y)

Using the rule sp(p, x := e) = [x0/x]p ∧ x = [x0/x]e,
([y0 / y]((y=0 ∧ x=4) ∧ x>y) ∧ y = [y0 / y]x) ∨ ((y=0 ∧ x=4) ∧ x≤y)

Using the substitution rule [e / x](e1 op e2) = [e / x]e1 op [e / x] e2,
((([y0 / y]y = [y0 / y]0 ∧ [y0 / y]x = [y0 / y]4) ∧ [y0 / y]x > [y0 / y]y) ∧
y = [y0 / y]x) ∨ ((y=0 ∧ x=4) ∧ x≤y)

Using the substitution rule [e / x]c = c,
((([y0 / y]y = 0 ∧ [y0 / y]x = 4) ∧ [y0 / y]x > [y0 / y]y) ∧ y = [y0 / y]x) ∨
((y=0 ∧ x=4) ∧ x≤y)

Using the substitution rule [e / x]y = y,
((([y0 / y]y = 0 ∧ x=4) ∧ x > [y0 / y]y) ∧ y=x) ∨ ((y=0 ∧ x=4) ∧ x≤y)

Using the substitution rule [e / x]x = e
(((y0=0 ∧ x=4) ∧ x>y0) ∧ y=x) ∨ ((y=0 ∧ x=4) ∧ x≤y)

55. wp(y:=x-1;z:=x+1, z>y)

Answer

wp(y:=x-1;z:=x+1, z>y)
=wlp(y:=x-1;z:=x+1, z>y) ∧ D(y:=x-1;z:=x+1)
=wlp(y:=x-1, wlp(z:=x+1, z>y)) ∧ D(y:=x-1) ∧ wp(y:=x-1, D(z:=x+1))
=wlp(y:=x-1, x+1>y) ∧ T ∧ wlp(y:=x-1, T) ∧ D(y:=x-1)
=wlp(y:=x-1, x+1>y) ∧ T ∧ wlp(y:=x-1, T) ∧ T

$=x+1>x-1$

56. $\mathrm{wlp}(z = x; y = x -1; x = x - y ; z < x)$

$= \mathrm{wlp}(z = x, \mathrm{wlp}(y = x -1; x = x - y ; z < x))$
$= \mathrm{wlp}(z = x, \mathrm{wlp}(y = x -1; \mathrm{wlp}(x = x - y ; z < x)))$
$= \mathrm{wlp}(z = x, \mathrm{wlp}(y = x -1; z < x - y))$
$= \mathrm{wlp}(z = x, z < x - (x - 1))$
$= \mathrm{wlp}(z = x, z < 1) = x < 1$

57. $\mathrm{wp}(a[j] := b[i] + a[i], a[j] > a[i])$
$= \mathrm{wlp}(a[j] := b[i] + a[i], a[j] > a[i]) \land D(a[j] := b[i] + a[i])$
$= ([b[i] + a[i]/a[j]]\ a[j] > a[i]) \land (0 \le j < |a| \land 0 \le i < |b| \land 0 \le i < |a|)$
$= (b[i] + a[i] > (i = j\ ?\ b[i] + a[i] : a[i])) \land (0 \le j < |a| \land 0 \le i < |b| \land 0 \le i < |a|)$
$\Leftrightarrow i = j\ ?\ (b[i] + a[i] > b[i] + a[i]) : (b[i] + a[i] > a[i]) \land (0 \le j < |a| \land 0 \le i < |b| \land 0 \le i < |a|)$
$\Leftrightarrow(i = j\ ?\ F : (b[i] > 0)) \land (0 \le j < |a| \land 0 \le i < |b| \land 0 \le i < |a|)$
$\Leftrightarrow i \neq j \land b[i] > 0 \land 0 \le j < |a| \land 0 \le i < |b| \land 0 \le i < |a|$

58. Find the strongest post condition:
$\mathrm{sp}(j \geq k \land m\ 6 = 5 \land k = 5, j := k * m; m := 5; k := j * m)$

Solution 1

$\mathrm{sp}(j \geq k \land m\ 6 = 5 \land k = 5, j := k * m; m := 5; k := j * m)$
$\mathrm{sp}(\mathrm{sp}(\mathrm{sp}(j_0 \geq k_0 \land m_0\ 6 = 5 \land k_0 = 5, j := k_0 * m_0), m:=5; k:=j*m))$
$\mathrm{sp}(\mathrm{sp}(j_0 \geq k_0 \land m_0\ 6 = 5 \land k_0 = 5 \land j = 5 * m_0, m:=5), k:=j*m))$
$\mathrm{sp}(j_0 \geq k_0 \land m_0\ 6 = 5 \land k_0 = 5 \land j = 5 * m_0 \land m = 5), k:=j*m))$
$= j_0 \geq k_0 \land m_0\ 6 = 5 \land k_0 = 5 \land j = 5 * m_0 \land m = 5 \land k = (5m_0) * 5$

59. $\mathrm{wp}(x = \mathrm{sqrt}(y) * (\frac{z}{k}), x>0)$

$= [\mathrm{sqrt}(y) * (\frac{z}{k}) / x]\ (x > 0) \land D(\mathrm{sqrt}(y) * (\frac{z}{k}))$
$= \mathrm{sqrt}(y) * (\frac{z}{k}) > 0 \land y \geq 0 \land k \neq 0$

60. $[\ x / y\ ]\ \forall x. x > y \land x > 0$

$= \forall x.\ [i/x\ ]x > y \land x > 0$
$= [\ x / y\ ]\ \forall i. i > y \land i > 0$
$= \forall i. i > x \land i > 0$

61. wp(if a[x] mod 2 = 1 then {a[x] := a[x] + 1} else {skip}, a[x] mod 2 = 0)
= wlp(if a[x] mod 2 = 1 then {a[x] := a[x] + 1} else {skip}, a[x] mod 2 = 0) ∧ D(if a[x] mod 2 = 1 then {a[x] := a[x] + 1} else {skip})
= ((a[x] mod 2 = 1) → wlp(a[x] := a[x]+1, a[x] mod 2 = 0)) ∧ (a[x] mod 2/= 1 → wlp(skip, a[x] mod 2 = 0)) ∧ D(a[x])
= (a[x] mod 2 = 1 → a[x] + 1 mod 2 = 0) ∧ (a[x] mod 2/= 1 → a[x] mod 2 = 0) ∧ x ≥ 0 ∧ x < |a|
(b) wp(a[k] := $\frac{a[k]}{2}$, a[k] = $2^{a[j]}$)
= wlp(a[k] := $\frac{a[k]}{2}$, a[k] = $2^{a[j]}$) ∧ D(a[k] := $\frac{a[k]}{2}$)
= [$\frac{a[k]}{2}$/a[k]](a[k] = $2^{a[j]}$) ∧ D(a[k])
= ($\frac{a[k]}{2}$ = [$\frac{a[k]}{2}$/a[k]]($2^{a[j]}$)) ∧ k ≥ 0 ∧ k < |a|
= $\frac{a[k]}{2}$ = $2^{(j=k\ ?\ \frac{a[k]}{2} : a[j])}$ ∧ k ≥ 0 ∧ k < |a|
= $\frac{a[k]}{2}$ = (j = k ? $2^{\frac{a[k]}{2}}$: $2^{a[j]}$) ∧ k ≥ 0 ∧ k < |a|

62. [x/y]∃x.(x * 2y >= z)

Solution:
[x/y]∃a.[a/x](x * 2y >= z)
=[x/y]∃a.(a * 2y >= z)
=∃a.(a * 2x >= z)

63. [x/y] ((∀x.∃y x>=y) → (∃y. z<=y))

Solution:
(∀x.∃y x>=y) → (∃y. z<=y)
There is no substitution required. Y is bound.

64. WLP

wlp(x:=x*y;skip,x=0)

wlp(x:=x*y,wlp(skip,x=0))
wlp(x:=x*y,x=0)
x*y=0

65. Let σ = {a = 5, b = 10, x = [5; 10; 15]}.

a) What is σ[a → 10][a → 5]?
SOLUTION:
= {a=10,b=10,x=[5;10;15]}[a→5]
= {a=5,b=10,x=[5;10;15]}

b) What is σ(x[2])?
SOLUTION:
=15
c) What is σ[a→ 4][b→ 10]?
SOLUTION:
= {a=4,b=10,x=[5;10;15]}[b→10]
= {a=4,b=10,x=[5;10;15]}

d) What is σ[c→ 1]
SOLUTION:
= {a=10,b=10,x=[5;10;15], c= 1}

66. For the following statements and states, give M(s, σ). Use ⊥d for divergence and ⊥e for errors; don't use just ⊥.
S = while(x>=1) {x=x2}, σ = { x= 1}
= {⊥d}
Because, the program will never leave the while loop as value of X will always be 1.

67. wp (a[i] := a[i] + 1, a[i] ≤ 5)

≡ [a[i] + 1/a[i]]a[i] ≤ 5
≡ a[i] ≤ 4

68.

wlp (a[i = 2 ? i : 2] := 20, a[j] = 30)

A2.
Let x = i = 2 ? i : 2
= [20/a[x]] (a[j] = 30)
= x = j ? 20 : a[j] = 30
= (j = i = 2 ? i : 2) ? 20 : a[j] = 30

69. wlp(a[i]=a[i+1]), a[even(x) ? i : j]

$= [a[i+1]/a[i]]\ (a[\mathrm{even}(x)\ ?\ i : j])$

$= (\mathrm{even}(x)\ ?\ i : j) = i ?\ a[i+1] : a[j]$

70.

$[x + y + 2/z](\exists x \forall y. z = x + y + 1)$

$= [x + y + 2/z](\exists x_0 \forall y_0. z = x_0 + y_0 + 1)$

$= (\exists x_0 \forall y_0. x + y + 2 = x_0 + y_0 + 1)$

$= \exists x_0 \forall y_0. x + y + 1 = x_0 + y_0$

71. $\mathrm{wlp}\ (a[x] := 1;\ \mathrm{if}\ x > 3\ \mathrm{then}\ a[x] := 2\ \mathrm{else}\ a[a[x]] := 3,\ a[x] \geq 3)$

$= \mathrm{wlp}\ (a[x] := 1,\ \mathrm{wlp}\ (\mathrm{if}\ x > 3\ \mathrm{then}\ a[x] := 2\ \mathrm{else}\ a[a[x]] := 3,\ a[x] \geq 3))$

$= \mathrm{wlp}\ (a[x] := 1,\ (x > 3 \rightarrow \mathrm{wlp}\ (a[x] := 2,\ a[x] \geq 3) \land x \leq 3 \rightarrow \mathrm{wlp}\ (a[a[x]] := 3,\ a[x] \geq 3))$

$= \mathrm{wlp}\ (a[x] := 1,\ (x > 3 \rightarrow 2 \geq 3 \land x \leq 3 \rightarrow [3/a[a[x]]]a[x] \geq 3)$

$= \mathrm{wlp}\ (a[x] := 1,\ (x > 3 \rightarrow F \land x \leq 3 \rightarrow x = a[x] ? 3 \geq 3 : a[x] \geq 3)$

$= \mathrm{wlp}\ (a[x] := 1,\ (x \leq 3 \lor F\ ) \land (x \leq 3 \rightarrow x = a[x] ? 3 \geq 3 : a[x] \geq 3)))$

$= \mathrm{wlp}\ (a[x] := 1,\ (x \leq 3) \land (x > 3 \lor x = a[x] ? 3 \geq 3 : a[x] \geq 3)))$

$= [1/a[x]](x \leq 3) \land (x > 3 \lor x = a[x] ? 3 \geq 3 : a[x] \geq 3))$

$= (x \leq 3) \land (x > 3 \lor x = 1 ? 3 \geq 3 : 1 \geq 3)$

$= (x \leq 3) \land (x > 3 \lor (x = 1 \land 3 \geq 3))$

$= (x \leq 3) \land (x > 3 \lor x = 1)$

$= (x \leq 3 \land x > 3) \lor (x \leq 3 \land x = 1)$

$= (x \leq 3 \land x = 1)$

Propositional and Predicate Logic

72. Use a truth table to prove the following expression as Tautology or contingency or contradiction

Expression: $(p \lor q) \land (\neg q \ \lor \ \neg r) => (p \land r)$

p	q	r	$p \lor q$	$\neg q$	$\neg r$	$\neg q \lor \neg r$	$(p \lor q) \land (\neg q \lor \neg r)$	$(p \land r)$	$(p \lor q) \land (\neg q \lor \neg r) => (p \land r)$
T	T	T	T	F	F	F	F	T	T
T	T	F	T	F	T	T	T	F	F
T	F	T	T	T	F	T	T	T	T
T	F	F	T	T	T	T	T	F	F
F	T	T	T	F	F	F	F	F	T
F	T	F	T	F	T	T	T	F	F
F	F	T	F	T	F	T	F	F	T
F	F	F	F	T	T	T	F	F	T

It is contingency.

73.

a) Complete the truth table for the proposition:
$(P \land (Q \to R)) \to (Q \to R)$

Ans:

P	Q	R	$Q \to R$	$P \land (Q \to R)$	$(P \land (Q \to R)) \to (Q \to R)$
T	T	T	T	T	T
T	T	F	F	F	T
T	F	T	T	T	T
T	F	F	T	T	T
F	T	T	T	F	T
F	T	F	F	F	T
F	F	T	T	F	T
F	F	F	T	F	T

b) Is $(P \land (Q \to R)) \to (Q \to R)$ a tautology, contradiction or contingency?

Ans:

From the above table, we see that the given proposition is a tautology as it is always true for all states of P, Q and R

74. For the following proportion (p ∨ q) ∧ (¬p) ∧ (¬q) .

A) draw the truth table for this statement?
B) Find whether the given statement is a tautology, contradiction or contingency?

Solution):-

A)The truth table

P	Q	(p ∨ q)	(¬p)	(¬q)	(p ∨ q) ∧ (¬p) ∧ (¬q)
T	T	T	F	F	F
T	F	T	F	T	F
F	T	T	T	F	F
F	F	F	T	T	F

C) form the above truth table to the given proportion logic we can say that

(p ∨ q) ∧ (¬p) ∧ (¬q) is the contradiction.

75. Proof

Proof the following logic implication, known as Disjunctive syllogism
((P ∨ Q) ∧ ¬P) ⇒ Q
Write a statement and a justification on each line, as you did on HW1. The justifications should be a
logic laws (See Appendix A) with a reference to the lines you're using. For example, if line 3 has the statemtn
P and line 4 has the statement Q, you could justify P ∧ Q with ”Conjunction (3, 4)”. If a justification uses
only one line and it's the line immediately before, you can leave it out. You can leave lines left over; there
may be more than you need.
1. ((P ∨ Q) ∧ ¬P) Assumption
2. ((P ∧ ¬P) ∨ (Q ∧ ¬P)) Distributivity
3. (F ∨ (Q ∧ ¬P)) Contradiction
4. (Q ∧ ¬P) Identity
5. Q Simplify

76. Construct a truth table for the proposition

(P→Q) ∧ R

P	Q	R	!P	P→Q	(P→Q) ∧ R
0	0	0	1	1	0
0	0	1	1	1	1
0	1	0	1	1	0
0	1	1	1	1	1
1	0	0	0	0	0
1	0	1	0	0	0
1	1	0	0	1	0
1	1	1	0	1	1

77. Check using truth tables if the following statement is true or not.

(P ∧ Q -> R) -> (~P ∨ ~Q -> ~R) ⇔ (~P ∨ ~Q -> ~R)

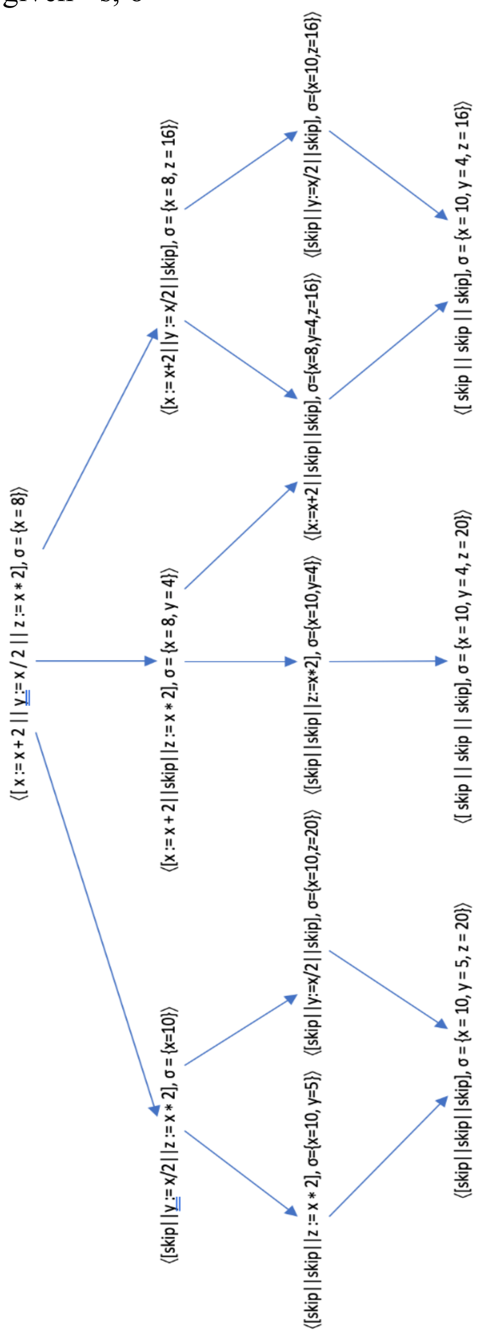
A1.

P	**Q**	**R**	**~P**	**~Q**	**~R**	**P^Q (A)**	**~P ∨ ~Q (~A)**	**A->R (X)**	**~A->~R (Y)**	**X->Y**
T	T	T	F	F	F	T	F	T	**T**	**T**
T	T	F	F	F	T	T	F	F	**T**	**T**
T	F	T	F	T	F	F	T	T	**F**	**F**
T	F	F	F	T	T	F	T	T	**T**	**T**
F	T	T	T	F	F	F	T	T	**F**	**F**
F	T	F	T	F	T	F	T	T	**T**	**T**
F	F	T	T	T	F	F	T	T	**F**	**F**
F	F	F	T	T	T	F	T	T	**T**	**T**

Parallel Programs

78. Consider s ≡ [x := x + 2 ‖ y := x / 2 ‖ z := x ∗ 2] and state σ = {x = 8}.

a) Draw an evaluation graph for given <s, σ>


⟨[x := x + 2 | | y := x / 2 | | z := x * 2], σ = {x = 8}⟩
⟨[x := x+2 | | y := x/2 | | skip], σ = {x = 8, z = 16}⟩
⟨[x := x + 2 | | skip | | z := x * 2], σ = {x = 8, y = 4}⟩
⟨[skip | | y := x/2 | | z := x * 2], σ = {x=10}⟩
⟨[skip | | y:=x/2 | | skip], σ={x=10,z=16}⟩
⟨[x:=x+2 | | skip | | skip], σ={x=8,y=4,z=16}⟩
⟨[skip | | skip | | z:=x*2], σ={x=10,y=4}⟩
⟨[skip | | y:=x/2 | | skip], σ={x=10,z=20}⟩
⟨[skip | | skip | | z := x * 2], σ={x=10, y=5}⟩
⟨[skip | | skip | | skip], σ = {x = 10, y = 4, z = 16}⟩
⟨[skip | | skip | | skip], σ = {x = 10, y = 4, z = 20}⟩
⟨[skip | | skip | | skip], σ = {x = 10, y = 5, z = 20}⟩

b) what is M(s, σ)?

Ans: M(s, σ) = {{x = 10, y = 5, z = 20}, { x = 10, y = 4, z = 20}, { x = 10, y = 4, z = 16}}

79. Draw an evaluation graph for <w, σ>. w ≡ [x := y*y; y := y+1 || await y < 2 then y := x+y] and σ = {x = 0, y= 1} and show if it has deadlock or not.

Let s≡await y < 2 then y := x+y

< [x := y*y; y := y+1 || await y < 2 then y := x+y], {x = 0, y= 1}>

↙ ↘

<[y := y+1|| s],{x=1,y=1}> <[x := y*y; y := y+1||skip],{x=0,y=1}>

↙ ↘ ↓

<[skip|| s], {x=1,y=2}> < [y := y+1||skip], {x=1,y=2}> < [y := y+1||skip], {x=1,y=1}>

(Deadlock) ↓ ↓

<[skip || skip], {x=1, y=3}> <[skip || skip], {x=1, y=1}>

It has a deadlock when we execute all expressions in thread 1 first. And then we can not execute thread 2 because y is not less than 2.

80. Compute M(w, σ) from previous problem.

M(w, σ) = {$\perp_d$, {x=1,y=3}, {x=1,y=1}

81. Consider s ≡ <[a := b + c || b := c/a || d:= 2]> and state σ = {b = 3, c = 12, d = 4}.

a) Draw an evaluation graph for <s, σ>.

b) What is M(s, σ)?

SOLUTION:

<[a := b + c || b := a/c], σ = {a=12, b = 18, c = 3}>

<[skip || b := a/c], σ[a→15] [b→18] [c→3]> <[a := b + c || skip], σ[a→12] [b→4] [c→3]>

<[skip || skip], σ[a→15][b→5] [c→3]> <[skip || skip], σ[a→7][b→4] [c→3]>

M(s, σ)= {{a=15,b=5,c=3},{a=7,b=4,c=3}}

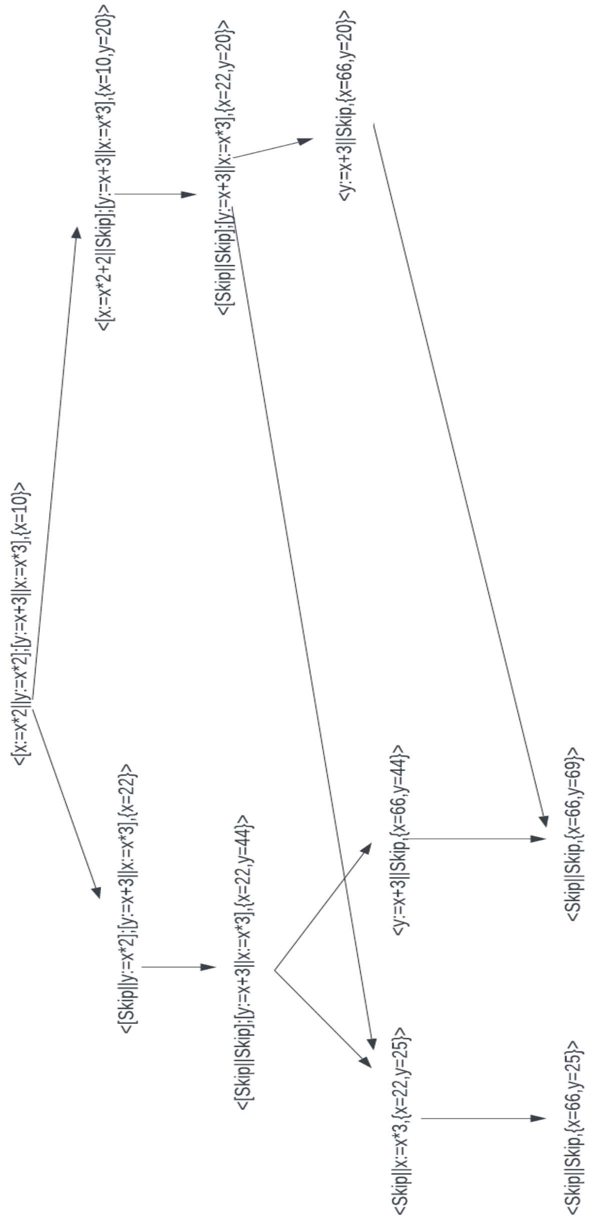
Draw the evaluation graph for the following:

S≡([x:=x*2+2 || y=x*2]; [y=x+3 || x=x*3]) and state σ={x=10}

b). What is M(s, σ)? You do not need to show a formal calculation, but explain briefly how you know.

Solution:

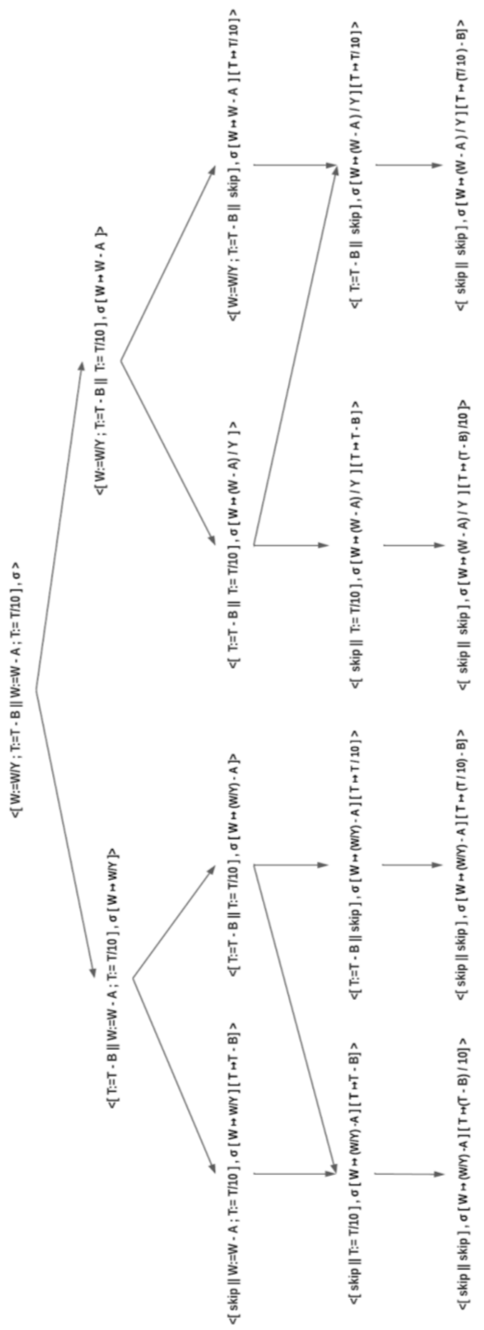
a). Next page


<[x:=x*2||y:=x*2];[y:=x+3||x:=x*3],{x=10}>
<[x:=x*2+2||Skip];[y:=x+3||x:=x*3],{x=10,y=20}>
<[Skip||Skip];[y:=x+3||x:=x*3],{x=22,y=20}>
<y:=x+3||Skip,{x=66,y=20}>
<[Skip||y:=x*2];[y:=x+3||x:=x*3],{x=22}>
<[Skip||Skip];[y:=x+3||x:=x*3],{x=22,y=44}>
<y:=x+3||Skip,{x=66,y=44}>
<Skip||Skip,{x=66,y=69}>
<Skip||x:=x*3,{x=22,y=25}>
<Skip||Skip,{x=66,y=25}>

b). M(s,σ)={{x=66,y=25},{x=66,y=69}}

82. Draw an evaluation graph for the following configuration R1 ≡ < S1,σ > where

S1 ≡ [W:=W/Y ; T:=T - B || W:=W - A ; T:= T/10]
What is M(s, σ)?


<[W:=W/Y ; T:=T - B || W:=W - A ; T:= T/10] , σ >
<[W:=W/Y ; T:=T - B || T:= T/10] , σ [W ↦ W - A]>
<[T:=T - B || W:=W - A ; T:= T/10] , σ [W ↦ W/Y]>
<[W:=W/Y ; T:=T - B || skip] , σ [W ↦ W - A] [T ↦ T/ 10] >
<[T:=T - B || T:= T/10] , σ [W ↦ (W - A) / Y] >
<[T:=T - B || T:= T/10] , σ [W ↦ (W/Y) - A]>
<[skip || W:=W - A ; T:= T/10] , σ [W ↦ W/Y] [T ↦ T - B] >
<[T:=T - B || skip] , σ [W ↦ (W - A) / Y] [T ↦ T/ 10] >
<[skip || T:= T/10] , σ [W ↦ (W - A) / Y] [T ↦ T - B] >
<[T:=T - B || skip] , σ [W ↦ (W/Y) - A] [T ↦ T / 10] >
<[skip || T:= T/10] , σ [W ↦ (W/Y) -A] [T ↦ T - B] >
<[skip || skip] , σ [W ↦ (W - A) / Y] [T ↦ (T/10) - B] >
<[skip || skip] , σ [W ↦ (W - A) / Y] [T ↦ (T - B) /10]>
<[skip || skip] , σ [W ↦ (W/Y) - A] [T ↦ (T / 10) - B] >
<[skip || skip] , σ [W ↦ (W/Y) -A] [T ↦(T - B) / 10] >

M(s, σ) =

{
σ [W ↦ (W/Y) -A] [T ↦(T - B) / 10] ,
σ [W ↦ (W/Y) - A] [T ↦ (T / 10) - B] ,
σ [W ↦ (W - A) / Y] [T ↦ (T - B) /10] ,
σ [W ↦ (W - A) / Y] [T ↦ (T/ 10) - B]
}

Hoare Triples

For this set of questions, say whether the triple is partially correct and totally correct.

83. {false} x :=x+5 {x≥0}

Solution:
Partially correct
Totally correct

84. {true} while x >0 { x:=x-1} {x =0}

Solution:
Partially not correct
Totally not correct

85. {true} while x ≤ 0 {x:=x-1} {x >0}

Solution:
Partially correct
Totally not correct

86. {x >0} if x ≥ 0 then x:= x - 1 else {skip} {x >0}

Solution:
Partially not correct
Totally not correct

87. {true} if x ≥ 0 then x:=x-1 else {skip} {x>0}

Solution:
Partially not correct
Totally not correct

88. Given a Hoare Triple as follows:

{n >0} S {y=2^n}

where

```
S = x:=0;  y := 1;  z:= 0;
        while x <n{x:=x+1;  z:=y;  y := y + z;}
```

(1) Give the loop Invariant for proof outline under partial correctness.

Solution:

inv p: **y=2^x ^ 0≤ x ≤n**

89. TRUE/FALSE

Check Validity for the following Triple

1. { p=p*2} s1 { q} = Invalid
2. {3>5} skip {False/True} = Valid

3. { x >= y } if x=y then x=y-x else x=x-y { x>=0} => Valid
4. {True} y=x*x*x { False} => Invalid

For the next two questions, is the triple valid?

90.[x = a] if(x > 0) then x := -x else {skip} [x = |a|]

It is not valid. (False)

91.{ x = a} if(x > 0) then {skip} else {x := -x} {x = |a|}

It is valid. (True)

92.Fill the precondition or Post condition

1. { } P = Q ; P= Q +Q { P=Q+Q }
   Ans: { T }
2. { } C= A + 3 ; B= C ;A=C+B {A>C /\ B=4}
   Ans: {A=1}

Language Semantics

93. Consider the following program, which we will refer to as S1 as first while loop and S2 as second while loop:

```
while (n>1){
        i:=0;
        while(a[i]>0){
                i:=a[i];
                }
        n:=a[i];
```

Show the steps of evaluation to evaluate s with the state σ = {a=[2,-1,0],n = 2} until it reaches skip. Do not skip over any steps that perform state updates.

Answer:

Assume first while will be S1 and second while will be S2.

→ <S1, {a= [2, -1,0], n=2}>

$\rightarrow^{2}$ <S2, n: =a[i]; S1; {a= [2, -1,0], n=2, i=0}>

→ <i: =a[i], n: =a[i]; S1; {a= [2, -1,0], n=2, i=0}> Because *σ(a[i] >0) =T*

→ <S2, n: =a[i]; S1; {a= [2, -1,0], n=2, i=2}>

$\rightarrow^{2}$ <n: =a[i]; S1; {a= [2, -1,0], n=2, i=2}> Because *σ(a[i] >0) =F*

→ < S1; {a= [2, -1,0], n=0, i=2}>

→ < skip; {a= [2, -1,0], n=0, i=2}> Because *σ(n >1) =F*