1. Consider the pair of predicate expressions:
$\forall x P(x) \land Q(x)$
$\forall x(P(x) \land Q(x))$
Are they the same or different? Briefly explain.
Answer:
$\forall x P(x) \land Q(x)$ means every x in the domain satisfy the property $P(x)$ and x has the property $Q(x)$ which may or may not be true.
$\forall x(P(x) \land Q(x))$ means every x in the domain satisfy both property $P(x)$ and $Q(x)$.
Therefore they are different.
2. Consider the pair of predicate expressions:
$\forall x \exists y P(x, y)$
$\exists y \forall x P(x, y)$
Are they the same, different, or does one imply the other? Use one or more examples to support your answer.
Answer:
They are different. For example, x, y are in the domain of integer and $P(x, y)\colon x = y$ then $\forall x \exists y P(x, y)$ would means for all x there exist a y such that $x = y$. $\exists y \forall x P(x, y)$ would means there exist a y such that, y equal to all x which is clearly false, so it is different to the first predicate expression.
3. Derive the weakest precondition wp in the following Hoare triple using inference rules. Show each step in your derivation.
$\{wp\}\ if\ (x > y)\ then\ (temp := x;\ x := y;\ y := temp;)\ \{x < y\}$
Answer:
By conditional rules, $wp(\ if\ (x > y) then\ (temp := x;\ x := y;\ y := temp;); x < y)$
$= (x > y \land wp(temp := x;\ x := y;\ y := temp; x < y)) \lor (\neg(x > y) \land x < y))$
By sequence rule, $wp(temp := x;\ x := y;\ y := temp; x < y)$
$= wp(temp := x;\ x := y;\ wp(\ y := temp; x < y))$
$= wp(temp := x;\ wp(x := y; wp(\ y := temp; x < y)))$
By assignment rule, $wp(temp := x;\ wp(x := y; wp(\ y := temp; x < y)))$
$= wp(temp := x;\ wp(x := y; x < temp))$
$= wp(temp := x;\ y < temp)$
$= y < x$
Substitute back, $wp(\ if\ (x > y) then\ (temp := x;\ x := y;\ y := temp;); x < y)$
$= (x > y \land\ y < x) \lor (\neg(x > y) \land x < y))$
$= x \neq y$
4. Use the inference rules for quantified statements in the following exercises.
(a) Prove $(\forall x P(x) \to \exists x Q(x)) \equiv \exists x(P(x) \to Q(x))$ Show each step in your proof.
Answer:
From the left hand side:
$\left(\forall x P(x) \to \exists x Q(x)\right) \equiv (\neg \forall x P(x)\ \lor\ \exists x Q(x))$ by Material implication
$(\neg \forall x P(x)\ \lor\ \exists x Q(x)) \equiv (\exists x\, \neg P(x)\ \lor\ \exists x Q(x))$ Since $\neg \forall x P(x)\ \equiv\ \exists x\, \neg P(x)$
$\left(\exists x\, \neg P(x) \lor\ \exists x Q(x)\right) \equiv\ \exists x\left(\neg P(x)\ \lor\ Q(x)\right)$ distributivity of exists
$\exists x\left(\neg P(x)\ \lor\ Q(x)\right)\ \equiv \exists x(P(x) \to Q(x))$ by Material implication
So that $(\forall x P(x) \to \exists x Q(x)) \equiv \exists x(P(x) \to Q(x))$.

(b) Translate the following description into predicate logic and prove correct.
People who drive cars have a driver's licence. John does not have a driver's licence.
Therefore some people don't drive cars.
Answer:
$\forall x(D(x) \to H(x)), \neg H(j)\ \vdash \exists x \neg D(x)$ where D stands for drive cars, H stands for have a driver's licence, x is in the domain of human and j is a person called John.

1	$\forall x\big(D(x) \rightarrow H(x)\big)$	promise
2	$\neg H(j)$	promise
3	$D(j) \rightarrow H(j)$	Universal Instantiation(1) with John since john ∈ domain of human
4	$\neg D(j)$	Modus tollens(2,3)
5	$\exists x \neg D(x)$	Existential Generalization(7)

(c) Write predicate formulae that formalise the following description and prove it correct.
Lies are statements that are both false and are known by the speaker to be false. Some false statements aren't known by the speaker to be false. Therefore, not all false statements are lies.

$\forall x(L(x) \rightarrow (F(x) \ \wedge \ K(x))), \exists x(F(x) \ \wedge \neg K(x)) \ \vdash \neg \forall x(F(x) \rightarrow L(x))$ while L stands for is lie, F stands for is false , K stands for known by speaker to be false.

1	$\forall x(L(x) \rightarrow F(x) \ \wedge \ K(x))$	Premise
2	$\exists x(F(x) \wedge \neg K(x))$	Premise
3	$\forall x(F(x) \rightarrow L(x)$	Assume conclusion incorrect
4	$F(a) \rightarrow L(a)$	Universal Instantiation(3) with constant a
5	$\exists x \neg\big(\neg F(x) \vee K(x)\big)$	De Morgan's Law(2)
6	$\exists x \neg\big(F(x) \rightarrow K(x)\big)$	Material implication(5)
7	$\neg \forall x\big(F(x) \rightarrow K(x)\big)$	Conversion of quantifiers(6)
8	$\neg\big(F(a) \rightarrow K(a)\big)$	Universal Instantiation(7) with constant a
9	$L(a) \rightarrow (F(a) \ \wedge \ K(a))$	Universal Instantiation(1) with constant a
10	$F(a) \rightarrow (F(a) \ \wedge \ K(a))$	Hypothetical syllogism(4,9)
11	$\neg F(a) \vee (F(a) \ \wedge \ K(a))$	Material implication(10)
12	$(\neg F(a) \vee F(a)) \ \wedge (\neg F(a) \vee \ K(a))$	Distributive laws(11)
13	$\mathrm{True} \wedge (\neg F(a) \vee \ K(a))$	$\big(\neg F(a) \vee F(a)\big)$ from(12) is tautology
14	$F(a) \rightarrow \ K(a)$	Material implication(13)
15	$\neg\big(F(a) \rightarrow K(a)\big) \ \wedge \ (F(a) \rightarrow \ K(a))$	Conjunction(8,14)
16	$\neg \forall x(F(x) \rightarrow L(x))$	Contradiction (15), assumption(3) incorrect

5. (a) Briefly describe what the program does

Answer:

Given the size of the array n, if the array A contain integer 0, the program stop execution with the index of the array r while A[r] == 0, else the program stops with r = -1.

(b) Briefly describe what the specification says.

Answer:

The specification says the precondition of the program is that n is a valid size of an array which is an integer greater than or equal to 1. The specification also says the postcondition of the program is that the program stops with integer r is either -1 or the valid array index, all the array variable checked which is array index greater than r is not equal to 0, and if the program stops with integer r is a valid array index then A[r] is equal to 0.

(c) Using the invariant:

$$r \ \in \ [-1, n \ - \ 1] \ \wedge \ (\forall i \ \in \ [r+1, n \ - \ 1](A[i] \neq 0))$$

prove that the code fragment is correct with respect to the specification

Answer:

To verify the program, we must check invariant established, invariant maintained, and loop termination since the program is basic a while loop.

Check invariant established :

Check if $\{n \ \geq \ 1\} \, r := \ n-1; \{\mathrm{r} \ \in \ [-1, \mathrm{n} \ - \ 1] \wedge \ (\forall \mathrm{i} \ \in \ [\mathrm{r}+1, \mathrm{n}-1](A[i] \neq 0))\}$ is true.

Execute the assignment statements on postcondition Rule 1

$\{n-1 \in [-1, \mathrm{n}-1] \wedge (\forall \mathrm{i} \in [\mathrm{n}, \mathrm{n}-1](A[i] \neq 0))\}$

$n-1 \in [-1, \mathrm{n}-1]$ means n must greater than or equal to 0 for $[-1, \mathrm{n}-1]$ to not be empty set

$\{n \geq 0 \wedge \text{true}\}$

Inference rule for precondition strengthening Rule 2

$$n \geq 1 \rightarrow n \geq 0, \qquad \{n \geq 0\} r := n-1; \{\mathrm{r} \in [-1, \mathrm{n}-1] \wedge (\forall \mathrm{i} \in [\mathrm{r}+1, \mathrm{n}-1](A[i] \neq 0))\}$$
$$\vdash \{n \geq 1\} r := n-1; \{\mathrm{r} \in [-1, \mathrm{n}-1] \wedge (\forall \mathrm{i} \in [\mathrm{r}+1, \mathrm{n}-1](A[i] \neq 0))\}$$

Hence invariant established.

Check invariant maintained :

Check if $\{\mathrm{r} \in [-1, \mathrm{n}-1] \wedge (\forall \mathrm{i} \in [\mathrm{r}+1, \mathrm{n}-1](A[i] \neq 0)) \wedge (r \neq -1 \wedge A[r] \neq 0)\} r := r-1;$
$\{\mathrm{r} \in [-1, \mathrm{n}-1] \wedge (\forall \mathrm{i} \in [\mathrm{r}+1, \mathrm{n}-1](A[i] \neq 0))\}$ is true.

Simplifying the precondition with Associative laws

$$\{\mathrm{r} \in [-1, \mathrm{n}-1] \wedge (\forall \mathrm{i} \in [\mathrm{r}+1, \mathrm{n}-1](A[i] \neq 0)) \wedge (r \neq -1 \wedge A[r] \neq 0)\}$$

Is the same as $\{\mathrm{r} \in [0, \mathrm{n}-1] \wedge (\forall \mathrm{i} \in [\mathrm{r}+1, \mathrm{n}-1](A[i] \neq 0)) \wedge A[r] \neq 0\}$

Is the same as $\{\mathrm{r} \in [0, \mathrm{n}-1] \wedge (\forall \mathrm{i} \in [\mathrm{r}, \mathrm{n}-1](A[i] \neq 0))\}$

Execute the assignment statements on postcondition Rule 1

$\{r-1 \in [-1, \mathrm{n}-1] \wedge (\forall \mathrm{i} \in [\mathrm{r}, \mathrm{n}-1](A[i] \neq 0))\}$

$\{r \in [0, \mathrm{n}] \wedge (\forall \mathrm{i} \in [\mathrm{r}, \mathrm{n}-1](A[i] \neq 0))\}$

Inference rule for precondition strengthening Rule 2

$$\{\mathrm{r} \in [0, \mathrm{n}-1] \wedge (\forall \mathrm{i} \in [\mathrm{r}, \mathrm{n}-1](A[i] \neq 0))\} \rightarrow \{r \in [0, \mathrm{n}] \wedge (\forall \mathrm{i} \in [\mathrm{r}, \mathrm{n}-1](A[i] \neq 0))\},$$
$$\{r \in [0, \mathrm{n}] \wedge (\forall \mathrm{i} \in [\mathrm{r}, \mathrm{n}-1](A[i] \neq 0))\} r := r-1; \{\mathrm{r} \in [-1, \mathrm{n}-1] \wedge (\forall \mathrm{i} \in [\mathrm{r}+1, \mathrm{n}-1](A[i] \neq 0))\}$$
$$\vdash \{\mathrm{r} \in [0, \mathrm{n}-1] \wedge (\forall \mathrm{i} \in [\mathrm{r}, \mathrm{n}-1](A[i] \neq 0))\} r := r-1;$$
$$\{\mathrm{r} \in [-1, \mathrm{n}-1] \wedge (\forall \mathrm{i} \in [\mathrm{r}+1, \mathrm{n}-1](A[i] \neq 0))\}$$

Hence invariant maintained.

Check loop termination:

Check if $\{\mathrm{r} \in [-1, \mathrm{n}-1] \wedge (\forall \mathrm{i} \in [\mathrm{r}+1, \mathrm{n}-1](A[i] \neq 0)) \wedge \neg(r \neq -1 \wedge A[r] \neq 0)\} \rightarrow$
$\{r \in [-1, n-1] \wedge (\forall i \in [r+1, n-1] A[i] \neq 0) \wedge (r \in [0, n-1] \rightarrow A[r] = 0)\}$ is true.

Simplifying the precondition with De Morgan's laws

$$\{\mathrm{r} \in [-1, \mathrm{n}-1] \wedge (\forall \mathrm{i} \in [\mathrm{r}+1, \mathrm{n}-1](A[i] \neq 0)) \wedge \neg(r \neq -1 \wedge A[r] \neq 0)\}$$

Is the same as $\{\mathrm{r} \in [-1, \mathrm{n}-1] \wedge (\forall \mathrm{i} \in [\mathrm{r}+1, \mathrm{n}-1](A[i] \neq 0)) \wedge (r = -1 \vee A[r] = 0)\}$

By Material implication, the result before is the same as

$$\{\mathrm{r} \in [-1, \mathrm{n}-1] \wedge (\forall \mathrm{i} \in [\mathrm{r}+1, \mathrm{n}-1](A[i] \neq 0)) \wedge (\neg(r \neq -1) \vee A[r] = 0)\}$$
$$\{\mathrm{r} \in [-1, \mathrm{n}-1] \wedge (\forall \mathrm{i} \in [\mathrm{r}+1, \mathrm{n}-1](A[i] \neq 0)) \wedge (r \neq -1 \rightarrow A[r] = 0)\}$$

It is obvious that

$$\{\mathrm{r} \in [-1, \mathrm{n}-1] \wedge (\forall \mathrm{i} \in [\mathrm{r}+1, \mathrm{n}-1](A[i] \neq 0)) \wedge (r \neq -1 \rightarrow A[r] = 0)\} \rightarrow$$
$$\{r \in [-1, n-1] \wedge (\forall i \in [r+1, n-1] A[i] \neq 0) \wedge (r \in [0, n-1] \rightarrow A[r] = 0)\}$$

Hence prove loop termination

Establish, maintain and termination are true, hence the program is verified.