1. The 2 expressions are the same. It does not matter that the forall statement is not wrapped around both statements in statement 1 as x is the only variable involved and the forall is implied for all x's in this statement
2. Predicate 2 implies predicate 1.
   For example, consider x being the domain of humans and y being the domain of pets. P(x, y) meaning that human x has pet y.
   - The first predicate says: for all humans x, there exists a pet y such that: human x has pet y. That is: all humans have at least 1 pet
   - The second predicate says: there exists a pet y such that for all humans x: human x has pet y. That is: there is a pet that all humans own (obviously physically impossible).

   If predicate 2 is true, every human has the same pet, so predicate 1 must be true as all humans have a pet. Therefore predicate 2 implies predicate 1 (2-> 1)

3. $\{wp\}\ if\ (x > y)\ then\ (temp := x;\ x := y;\ y := temp;)\ \{x < y\}$
   $wp(if\ (x > y)\ then\ (temp := x;\ x := y;\ y := temp;)\ x < y)$
   $= (x > y) \wedge \mathrm{wp}(temp := x;\ x := y;\ y := temp;\ x < y)\ \vee\ (\neg(x > y) \wedge (x < y))$
   $= (x > y) \wedge \mathrm{wp}(temp := x;\ x := y;\ wp(y := temp;\ x < y))\ \vee\ (\neg(x > y) \wedge (x < y))$
   $= (x > y) \wedge \mathrm{wp}(temp := x;\ x := y;\ \ x < temp)\ \vee\ (\neg(x > y) \wedge (x < y))$
   $= (x > y) \wedge \mathrm{wp}(temp := x;\ wp(x := y;\ \ x < temp))\ \vee\ (\neg(x > y) \wedge (x < y))$
   $= (x > y) \wedge \mathrm{wp}(temp := x;\ y < temp)\ \vee\ (\neg(x > y) \wedge (x < y))$
   $= (x > y) \wedge (y < x)\ \vee\ (\neg(x > y) \wedge (x < y))$
   $= (x > y) \wedge (y < x)\ \vee\ ((x \leq y) \wedge (x < y))$
   $= (x > y) \wedge (y < x)\ \vee (x < y))$
   $= (x > y)\ \vee (x < y))$
   $= x\ \neq\ y$
   Therefore, the weakest precondition $=\ x\ \neq\ y$
4. Proof
   a. $(\forall \mathrm{xP(x)} \rightarrow\ \exists \mathrm{xQ(x)}) \equiv\ \exists \mathrm{x}(\mathrm{P(x)} \rightarrow\ \mathrm{Q(x)})$
      $LHS =\ (\forall \mathrm{xP(x)} \rightarrow\ \exists \mathrm{xQ(x)})$
      $=\ (\neg(\forall \mathrm{xP(x)})\ \vee\ \exists \mathrm{xQ(x)})$ – Material Implication
      $=\ (\exists \mathrm{x}\neg \mathrm{P(x)}\ \vee\ \exists \mathrm{xQ(x)})$ – Conversion of quantifiers (DeMorgan's Law)
      $= \exists \mathrm{x}(\neg \mathrm{P(x)}\ \vee\ \mathrm{Q(x)})$ – Distributivity of exists
      $= \exists \mathrm{x}(\mathrm{P(x)}\ \ \rightarrow\ \ \mathrm{Q(x)})$ – Material Implication
      $=\ RHS$

   b. People who drive cars have a driver's licence. John does not have a driver's licence. Therefore, some people don't drive cars.
      $\forall x(Drives(x) \rightarrow\ Licence(x)), \exists x(\neg Licence(x))\ \vdash\ \exists x(\neg Drives(x))$

1	$\forall x(Drives(x) \rightarrow\ Licence(x)$	Premise
2	$\exists x(\neg Licence(x))$	Premise
3	$\neg Licence(a)$	Existential Instantiation (2)
4	$\neg Drives(a)$	Modus Tollens (1, 3)
5	$\exists x(\neg Drives(x))$	Existential Generalisation (4)

   c. 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.

$\forall x((False(x) \wedge \text{SpeakerKnowsFalse(x)}) \rightarrow Lie(x))$,
$\exists x\big(False(x) \wedge \neg\text{SpeakerKnowsFlase(x)}\big)$,
$\vdash\ \neg\forall x(False(x)\ \rightarrow \text{Lie(x)})$

1	$\forall x\big(False(x)$ $\wedge \text{SpeakerKnowsFalse(x)}\big)$ $\rightarrow \big(Lie(x)\big)$	Premise
2	$\exists x\big(False(x)$ $\wedge \neg\text{SpeakerKnowsFalse(x)}\big)$	Premise
3	$(False(a)\ \wedge \neg SpeakerKnowsFalse(a))$	Existential instantiation (2)
4	$\exists x(False(a)\ \wedge \neg SpeakerKnowsFalse(a))$	Existential generalisation (3)
5	$\neg\forall x\big(False(x)$ $\wedge \neg\text{SpeakerKnowsFalse(x)}\big)$ $\rightarrow \big(Lie(x)\big)$	Modus Tollens (1, 4)
6	$\neg\forall x(False(x)\ \rightarrow \text{Lie(x)})$	Conjunction elimination (5)

5. Code fragment
    a. This code iterates through an array (A) using index r from highest index (n-1) to lowest index (0). If an element of the array at index r is 0 or the index (r) is -1, the loop is not executed and the end of the fragment is reached
    b. Precondition required is n >= 1 (size of array must be at least 1), Postcondition requires that r is between -1 and n – 1 (size of the array) and for every r value between the final r value + 1 and n – 1 (size of array) A[r] is not equal to 0 and if the final value of r was between 0 and n – 1 then there is a value for r between 0 and n – 1 where A[r] = 0
    c. Invariant = $r\ \in\ [-1, n\ -\ 1]\ \wedge\ (\forall i\ \in\ [r+1, n\ -\ 1](A[i]\ !=\ 0))$
    $\text{B} =\ \text{r}\ !=\ -1\ \wedge\ \text{A[r]}\ !=\ 0$
    $!B = (0 \leq r \leq (n-1)) \rightarrow\ A[r] = 0$
    $I\ \&\&\ !B\ =\ \{\text{r}\ \in\ [-1, \text{n}\ -\ 1]\ \wedge\ (\forall \text{i}\ \in\ [\text{r}+1, \text{n}\ -\ 1]\text{A[i]}\ !=\ 0)\ \wedge\ (\text{r}\ \in\ [0, \text{n}\ -\ 1]\ \rightarrow\ \text{A[r]}\ =\ 0)\}$
    = Postcondition