CSCI 2600 — Principles of Software

Homework 1: Reasoning About Code

Due: Monday, Feb. 3, 2025, 11:59:59 pm

Submission Instructions

- This homework is different from the previous homework. It does not involve any Java coding. Rather, you will answer some questions involving reasoning about code.
- Follow the directions in the version control handout for cloning your hw01 git repo.The URI that you will need to use to clone your personal repo for this homework would have the form of `https://submitty.cs.rpi.edu/git/s25/csci2600/hw01/RCSID` where `RCSID` is your RCS ID. Submit your answers in a single .PDF file named hw1_answers.pdf in the answers/ directory of your repository. **You MUST type up your answers. Handwritten solutions will not be accepted or graded, even if they are scanned into a PDF file.** We recommend using LaTeX. If you have never used LaTeX, take a look at this tutorial.
- Be sure to commit and push the file to Submitty. Follow the directions in the version control handout for adding and committing files.
- You must click the **Grade My Repository** button for you answers to be graded. If you do not, they will not be graded and you will receive a zero for the homework.

Directions:

- Unless otherwise indicated, assume that all referenced numerical variables are defined as integers, and that integer overflow or underflow will not occur.
- In your answers, please use standard logical symbols $\wedge$ and $\vee$ for "and" and "or" respectively. Use = for equivalence (comparing equality) in logical statements.
- Note that braces ({ and }) are used in this homework to enclose conditions, as well as to define blocks of code. Make sure you distinguish between these two different uses of braces based on the context.
- If no precondition is required for a code sequence, simply write `{␣true␣}` to denote the trivial precondition.
- You must **show work** for all problems in this homework in order to get credit. If you provide just the final answer but fail to show work, you will get a zero, even if the answer is correct.
- All code is in Java and assumes the use of standard Java library.

Problems

Problem 1 (12 pts., 6 pts. each part): Condition Strength

Part (a). State whether the following implication chains are true. Note that the implication arrow in this part is pointing to the left, i.e., the rightmost condition would be the strongest, if the chain is true. For any chains that are not true, indicate all implications that make those chains false, and provide a counterexample for each false implication.

$|\cdot|$ is the absolute value operator.

E.g., this chain { `x > 0` } ← { `x is odd` } ← { `x is prime` } ← { `x = 37` } is false because the following implications are false:

- { `x is odd` } ← { `x is prime` }. Counterexample: $x = 2$ which makes this implication { `false` } ← { `true` } which is `false`.
- { `x > 0` } ← { `x is odd` }. Counterexample: $x = -1$ which makes this implication { `false` } ← { `true` } which is `false`.

(1) { `x is a Monday` } ← { `x is a day in January 2025` } ←
← { `x is a Monday in January` } ← { `x is Monday, January 27, 2025` }

(2) { `u = 10k` ∧ `v = y + 7` ∧ `u + v = 7k` } ← { `10x + y is divisible by 7` } ←
← { `x - 2y is divisible by 7` ∨ `x - 2y = 0` } ← { `x = 60` ∧ `y = 9` } ←
← { `10x + y = 609` }

(3) { `true` } ← { $|x| =$ `x` } ← { `x > 0` } ← { `10 < x < 50` ∧ `x < 0` } ← { `false` }

Part (b). Order conditions from **the weakest to the strongest** by showing the longest implication chain of the form A ← B ← C ← D ... (where A, B, C, D, ... are given conditions, ← is implication). If total ordering by strength cannot be established, show a partial ordering. Prove by showing all necessary implications. If any implication is false, provide a counterexample. You must show all work in order to get full credit. Unless otherwise stated, assume all referenced variables are defined as integers.

(1) { $6 \le$ `k` $< 6 \wedge$ `k = 6` }
{ `false` }
{ k = 6n }
{ $12 \le$ k ≤ 12 }

(2) { `x = 3k + 3` ∧ `x is even` }
{ x is divisible by 6 }
{ y = the sum of digits of x ∧ y % 3 = 0 ∧ x = 2k }
{ x = x + 1 ∧ y = 12 }

(3) Assume `result` is a double.
{ `result` = $\sqrt{x}$ }
{ `|result`^{2} `- x|` $\ge$ `0.0` }
{ `|result`^{2} `- x|` $\le$ `-10`$^{-10}$ }
{ `|result`^{2} `- x|` $\le$ `0.001` }

Problem 2 (4 pts., 1 pt. each): Hoare Triples

State whether each Hoare triple is valid. If it is invalid, explain why and show how you would modify the postcondition to make it the strongest for the given precondition. Unless otherwise stated, assume all referenced variables are defined as integers.

(1) { x $\geq -\frac{1}{2}$ }

```
y = 2 * x;
```

{ y $\geq$ 0 $\vee$ y = 1 }

(2) { $\sqrt{x-1} \leq$ k }

```
x = x - 1;
```

{ k $<$ 0}

(3) { i + j $\neq$ 0 $\wedge$ i $\cdot$ j = 0}

```
i = j + 1;
k = i * j;
j = i - 1;
```

{ i = 0 $\vee$ i = j $\vee$ k = i $\cdot$ j }

(4) { false }

```
if (m < n)
  x = n;
else
  x = m;
```

{ x = min(n, m) }

Problem 3 (4 pts., 1 pt. each): General Hoare Triples

A, B, C, D, E, and F are logical conditions (logical formulas). The following are true:

- A → B (A implies B, i.e., A is stronger than B)
- B → C
- C → D
- D → E
- E → F
- E → G
- { B } code { E }

Indicate whether the following Hoare triples are valid or possibly invalid. If possibly invalid, prove by giving an example.

(1) {E} code {B}

(2) {B} code {E}

(3) {C} code {D}

(4) {E} code {E}

Problem 4 (8 pts., 1.5 pts. for each condition): Forward reasoning

For each code snippet with the given precondition, find the **strongest postcondition** by inserting the appropriate condition in each blank. The first intermediate condition in part (1) is supplied as an example. Please simplify your answers as much as possible. Assume all referenced variables are defined as integers. Copy all code to your answer file and fill in the blanks. Carry all variables forward. Show all work.

(1)
```
{ z = 0 }
x = 10;
    { x = 10 ∧ z = 0}
y = y - x;
   {_____________________________________________________}
z = x - y;
   {_____________________________________________________}
y = 0;
   {_____________________________________________________}
z = 2 * k;
   {_____________________________________________________}
```

(2)
```
{ |x| > 4 }
y = x;
   {_____________________________________________________}
x = -x * y;
   {_____________________________________________________}
x = x + y;
   {_____________________________________________________}
```

(3)
```
{ xy = 0 }
if (x > 0 || y > 0) {
     {_____________________________________________________}
   y = y * x;
     {_____________________________________________________}
} else {
     {_____________________________________________________}
   x = x + y;
     {_____________________________________________________}
}
   {_____________________________________________________}
```

Problem 5 (12 pts., 0.5 pts. each condition): Backward reasoning

Find the **weakest precondition** of each code sequence by inserting the appropriate condition in each blank. The first intermediate condition in part (1) is supplied as an example. Please simplify your answers as much as possible. Assume all referenced variables are defined as integers.

(1) Use the `wp()` notation shown in the example for all conditions in this part.

```
   {_____________________________________________________}
x = x + k;
    { wp("z = 2 * y + x;", z ≤ 10) } = { 2y + x ≤ 10 } =
    { 2y ≤ 10 - x } = { y ≤ (10−x)/2 }
z = 2 * y + x;
   { z ≤ 10 }
```

(2) Use the `wp()` notation for all conditions in this part.

```
   {_____________________________________________________}
y = x;
   {_____________________________________________________}
x = -x * y;
   {_____________________________________________________}
x = x + y;
   {_____________________________________________________}
   { x > 0 }
```

(3)
```
{_____________________________________________________}
if (y >= 10) {
     {_____________________________________________________}
    x = y / 10; // integer division
     {_____________________________________________________}
    y = y % 10;
     {_____________________________________________________}
} else {
     {_____________________________________________________}
    y = x;
     {_____________________________________________________}
}
  { x < 0 ∧ y ≤ -10 }
```

(4)
```
{_____________________________________________________}
if (Math.abs(x) <= 5) {
     {_____________________________________________________}
    x = x + 2;
     {_____________________________________________________}
} else {
     {_____________________________________________________}
  if (x <= -5) {
       {_____________________________________________________}
    z = x + 6;
       {_____________________________________________________}
  } else {
       {_____________________________________________________}
    x = 2 * z;
       {_____________________________________________________}
  }
     {_____________________________________________________}
}
  { -5 ≤ x ≤ 2 ∧ z < 0 }
```

(5)
```
{_____________________________________________________}
if (x < 10) {
     {_____________________________________________________}
    z = z < 0 ?  Math.max(z, x):x + 10;
     {_____________________________________________________}
    x = x + y;
     {_____________________________________________________}
}
{ z ≠ 0 ∧ y ≥ 0 ∨ x ≥ 0}
```

Problem 6 (5 pts., 0.5 pts. each condition, 0.5 pts. sufficient/insufficient): Verifying Correctness

For each block of code, fill in all the conditions, then use them to state whether the precondition is sufficient to guarantee the postcondition. If the precondition is insufficient, explain why.

Hint: Use backward reasoning to find the weakest precondition that guarantees the postcondition and see if the given precondition is strong enough to guarantee the postcondition. In other words, is the given precondition not stronger than the weakest precondition?

Copy all code to your answer file and fill in the blanks. Show all work. Assume all referenced variables are defined as integers.

(1)
```
{ x < 2 }
   {_____________________________________________________________}
z = x * 2;
   {_____________________________________________________________}
w = -z;
   {_____________________________________________________________}
w = w - 1;
{w > 1}
```
Sufficient or Insufficient: _____

(2)
```
   { x = y ∧ y > 0 ∨ y ≠ x }
   {_____________________________________________________________}
if (x == y) {
     {_____________________________________________________________}
  x++;
     {_____________________________________________________________}
} else {
     {_____________________________________________________________}
  x = y + 2;
     {_____________________________________________________________}
}
   { x > y ∧ y > 0}
```
Sufficient or Insufficient: _____

Problem 7 (4 pts.): Finding Input Values

Find all possible values of inputs that cause the code below to produce the output observed. Assume all variables are ints and have been properly declared and initialized. You need to apply a reasoning technique, not just "see" or "guess" the answer or run experiments with the code. Remember that you need to find **all** combinations of inputs, not just one. Show all work.

```
if (x >= y - b) {
    y = y + b * x;
    b = b + x + y;
} else {
    b = 1 - x;
    x = y - x;
}
System.out.printf("%b %b %b\n", b < 0, x > y, y < 0);
```

prints

```
true false true
```

Collaboration (0.5 pts)

Please answer the following questions in a file named `hw1_collaboration.pdf` in your `answers/` directory.

The standard academic integrity policy applies to this homework.

State whether or not you collaborated with other students. If you did collaborate with other students, state their names and a brief description of how you collaborated.

Reflection (0.5 pts)

Please answer the following questions in a file named `hw1_reflection.pdf` in your `answers/` directory. Answer briefly, but in enough detail to help you improve your own practice via introspection and to enable me to improve Principles of Software in the future.

- In retrospect, what could you have done better to reduce the time you spent solving this homework?
- What could we, the teaching staff, have done better to improve your learning experience in this homework?
- What do you know now that you did not know before beginning the homework?

  We will be awarding up to 1 extra credit point (at the discretion of the grader) for particularly insightful, constructive, and helpful reflection statements.

Errata

Check the Submitty Forum for possible errata.