Problem Set 6

1. (20 points) Our goal in this problem is to prove the correctness of the Hoare triple $\{n > 0\}$ S $\{y = n{\times}n\}$ where S is the following program:

```
y := 0; i:=0;
while(i<n) {
  t := 2i+1;
  y := y+t;
  i := i+1;
}
```

   (a) (4 points) State an inductive loop invariant I that is sufficient to prove the correctness of the above Hoare triple.

   (b) (7 points) Compute the weakest precondition of I (from part (a)) with respect to the loop body B.

   (c) (9 points) Show all VCs that are generated for proving the Hoare triple $\{n > 0\}$ S $\{y = n \times n\}$ using invariant I from part (a).

2. (15 points) Consider the following proposed proof rule to be added to Hoare logic:

$$\frac{\vdash \{P\}S\{Q\}}{\vdash \{P \land R\}\ S\ \{Q \land R\}}$$

   where R represents any formula.

   (a) (3 points) Prove that this rule is unsound.

   (b) (4 points) Under what restrictions on S would the above rule be sound?

   (c) (8 points) Prove that your modified rule from part (b) is now sound.

   Note: You must explicitly state any assumptions you make about S.

3. (10 points) Consider a (side-effect-free) function `F` with arguments `x1`, ... `xn` and suppose that `F` has precondition P and post-condition Q (over variable `ret`). Now, consider the following call-site of `F`:

```
x := F(e1, ..., en);
```

   Is it sound to model this callsite with the following code snippet?

```
assert(P[e1/x1, ... en/xn]);
assume(Q[x/ret, e1/x1, ... en/xn]);
```

   If so, argue why this is correct; otherwise, give a counterexample to illustrate why this is unsound.

4. (25 points) In this question, we will explore the interval abstract domain in a bit more detail.

(a) (5 points) Recall that an *abstract transformer* for a statement yields the new abstract values for program variables given their old abstract value. What are the abstract transformers for the statements `assume`$(x \leq c)$ and `assume`$(x > c)$ assuming that x's initial abstract value is $[l, u]$ and c is an integer constant?

(b) (5 points) Consider the following program:

```
0:
1:      x = 1;
2:
3:      while(x<1000) {
4:
5:      x:= x+1;
6:
7:      if(x>99) break;
8:
9:    }
10:   assert(x == 100);
```

Suppose that we model the statement `if(c) S1 else S2` as:

`if(*) { assume(c); S1; } else { assume(!c); S2; }`

and similarly for `while` statements. Show the control flow graph for the above program under this assumption.

(c) (5 points) What are the abstract values for x at program locations labeled (4) and (10) after 3 iterations of fixed point computation using the interval abstract domain?

(d) (5 points) What are the abstract values for x at program locations labeled (4) and (10) after applying widening to the result from part (c)? Can the assertion at line (10) be proven?

(e) (5 points) What are the abstract values for x at program locations labeled (4) and (10) after applying narrowing to the result from part (d)? Can the assertion at line (10) be proven now?