

Program Analysis and Verification

Assignment 4

(Due on 23rd October 2007)

1. Consider the Hoare rule for **while**.

$$\frac{\vdash \{A\}c\{b \Rightarrow A \wedge \neg b \Rightarrow B\}}{\vdash \{b \Rightarrow A \wedge \neg b \Rightarrow B\}\mathbf{while} \ b \ \mathbf{do} \ c\{B\}}$$

Derive this using the rules presented in the lecture.

2. Consider the following Hoare rules for **while**.

$$\frac{\vdash \{A\}c\{A\}}{\vdash \{A\}\mathbf{while} \ b \ \mathbf{do} \ c\{A \wedge \neg b\}} \text{ Rule 1} \quad \frac{\vdash \{A \wedge b\}c\{A\}}{\vdash \{A\}\mathbf{while} \ b \ \mathbf{do} \ c\{A\}} \text{ Rule 2}$$

Prove that the rules 1 and 2 are not complete, i.e. show the existence of A, B, c such that $\models \{A\}c\{B\}$, but $\vdash \{A\}c\{B\}$ cannot be derived using either rule 1 or rule 2.

3. In the derivation for computing WP for **while**, we saw that

$$WP(\mathbf{while} \ \dots, B) = \bigwedge F^i(\text{true})$$

where

$$F(A) = b \Rightarrow WP(c, A) \wedge \neg b \Rightarrow B$$

Prove that F is monotone and continuous.

4. Define a natural partial order \sqsubseteq on the set of assertions S defined in the class and prove that (S, \sqsubseteq) is a complete lattice.
5. Use ESCJava2 to verify the sort method after annotating the code given below with appropriate preconditions, postconditions, class invariant and loop invariant.

```
public class Sort {  
  
    private int A[];  
    private int cur;  
  
    public Sort(int sz){  
        A = new int[sz];  
        cur = 0;  
    }  
  
    public void addElement(int el){
```

```

    A[cur++] = e1;
}

public void sort(){
    int i, j;
    for(i = 0; i < cur-1; i++)
        for(j = i+1; j < cur; j++)
            if (A[j] < A[i])
                swap(i,j);
}

public void swap(int i, int j) {
    int tmp = A[i];
    A[i] = A[j];
    A[j] = tmp;
}
}

```

You must submit the annotated source file Sort.java. No warnings must be produced when the following command is executed.

```
escj Sort.java
```

6. Explain why ESCJava2 behaves differently for the two methods with identical implementations shown below.

```

public class Loop {

    //@ requires i >= 5;
    public static void infiniteLoop1(int i) {
        //@ loop_invariant i >= 5;
        while(i >= 0) i = 5;
        //@ assert false;
    }

    public static void infiniteLoop2(int i) {
        while(i >= 0) i = 5;
        //@ assert false;
    }
}

```

7. Use ESCJava2 to verify the search method after annotating the code given below with appropriate preconditions, postconditions, class invariant and loop invariant and correcting logical errors if any.

```

public class Search {
    private int A[];

    private int cur;

    public Search(int sz){
        A = new int[sz];
        cur = 0;
    }

    public void add(int el){
        A[cur++] = el;
    }

    //if x exists in array A, then return true.
    //Otherwise, return false.
    public boolean search(int x) {
        int i = 0;
        boolean present = false;

        while(i < cur-1 && !present) {
            if(A[i] == x)
                present = true;
            i++;
        }

        return present;
    }
}

```

You must submit the annotated source file Search.java. No warnings must be produced when the following command is executed.

```
escj Search.java
```