

Week 10 Tutorial  
Weakest Precondition Calculus

---

### Question 1

Let  $C$  be the program fragment:

```
if (x>y) then
  tmp:=x;
  x:=y;
  y:=tmp
end
```

Express the following in as simple a form as you can:

$$wp(C, (x = 3 \wedge y = 5))$$

### Question 2

Let  $B$  be the program fragment:

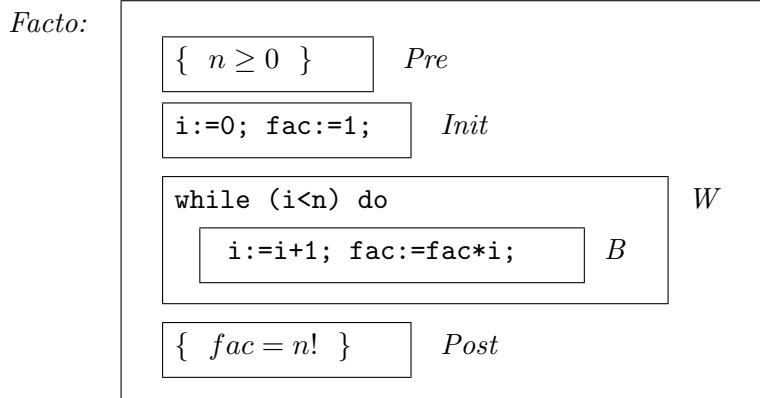
```
if (k>0) then
  k:=k-1;
  n:=n*2
end
```

Express the following in as simple a form as you can:

- a)  $wp(B, (k = 0 \wedge n = 0))$
- b)  $wp(B, (n = 2^{j-k}))$

### Question 3

The following is a program *Facto* (decorated with assertions) for computing the factorial of a positive integer.



We use *Pre* to stand for the precondition, *Init* to stand for the initialization code, *B* to stand for the body of while statement *W* and *Post* to be the program postcondition. We define a loop postcondition *Q* which is  $(fac = i! \wedge 0 \leq i = n)$ .

Finally, we define  $P_k$  to be the predicate that must be true before *W* executes to guarantee that the loop terminates after exactly *k* iterations in a state that satisfies *Q*.

1. Give expressions for  $P_0, P_1, P_2$  in as simple a form as you can.
2. Infer an expression for  $P_k$ . (Don't worry about the inductive proof.)
3. Give an expression for  $wp(W, Q)$  in as simple a form as you can.
4. Verify that  $wp(Facto, Q) \equiv n \geq 0$ .
5. Show that the program is correct with respect to its specification. That is,  $\{Pre\} Facto \{Post\}$ .

## 1 Appendix: Weakest Precondition Rules

$$wp(x := e, Q(x)) \equiv Q(e)$$

$$wp(S_1; S_2, Q) \equiv wp(S_1, wp(S_2, Q))$$

$$\begin{aligned} wp(\text{if } b \text{ then } S_1 \text{ else } S_2, Q) &\equiv (b \rightarrow wp(S_1, Q)) \wedge (\neg b \rightarrow wp(S_2, Q)) \\ &\equiv (b \wedge wp(S_1, Q)) \vee (\neg b \wedge wp(S_2, Q)) \end{aligned}$$

$$\begin{aligned} wp(\text{if } b \text{ then } S, Q) &\equiv (b \rightarrow wp(S, Q)) \wedge (\neg b \rightarrow Q) \\ &\equiv (b \wedge wp(S, Q)) \vee (\neg b \wedge Q) \end{aligned}$$

$P_k$  is the weakest predicate that must be true before `while b do S` executes, in order for the loop to terminate after exactly  $k$  iterations in a state that satisfies  $Q$ .

$$P_0 \equiv \neg b \wedge Q$$

$$P_{k+1} \equiv b \wedge wp(S, P_k)$$

$$wp(\text{while } b \text{ do } S, Q) \equiv \exists k. (k \geq 0 \wedge P_k)$$