8 Hoare Logic and Model Checking (jp622)

Consider commands C composed from assignments $X := E$ (where X is a program variable, and E is an arithmetic expression), heap dereference $X := [E]$, heap assignment $[E_1] := E_2$, the no-op `skip`, sequencing $C_1; C_2$, conditionals $\mathtt{if}\ B\ \mathtt{then}\ C_1\ \mathtt{else}\ C_2$ (where B is a boolean expression), and loops $\mathtt{while}\ B\ \mathtt{do}\ C$. `null` is 0. We write $\mathtt{align}(t, s)$ for the smallest multiple of s larger than t.
Let $\mathsf{block}(t, 0) = \mathsf{emp}$, $\quad \mathsf{block}(t, n + 1) = (\exists t'.\, t \mapsto t') * \mathsf{block}(t + 1, n)$.

(*a*) Explain why the following postcondition for an allocator that returns aligned blocks is incorrect, and propose a fix.

$$\begin{array}{l} \{\mathsf{block}(B, E - B) * 1 \le S\} \\ \mathtt{if}\ \mathtt{align}(B, 2^S) + 2^S < E \\ \mathtt{then}\ (R := \mathtt{align}(B, 2^S); B := B + 2^S)\ \mathtt{else}\ R := 0 \\ \left\{ \begin{array}{l} \mathsf{block}(B, E - B) * \\ (R \neq 0 \Longrightarrow (\mathsf{block}(R, 2^S) * R = \mathtt{align}(R, S))) \end{array} \right\} \end{array}$$

[3 marks]

(*b*) With this specification, allocations cannot be chained, as in $C_{\text{alloc}}; Y := X; C_{\text{alloc}}$. Explain why, and propose a fix. [2 marks]

(*c*) Strengthen the precondition just enough to guarantee the success of allocation (so that $R \neq 0 \Longrightarrow$ is not needed anymore). [2 marks]

(*d*) Consider the following representation predicate for lists of free blocks of size 2^S:

$$\mathtt{freelist}(t, S) = (t = \mathtt{null} * \mathtt{emp}) \lor \left(\exists t'. \left(\begin{array}{l} t = \mathtt{align}(t, 2^S) \land \\ \left(\begin{array}{l} t \mapsto t' * \\ \mathtt{block}(t + 1, 2^S - 1) * \\ \mathtt{freelist}(t', S) \end{array} \right) \end{array} \right) \right)$$

Give a loop invariant, and precisely but informally explain why it is preserved, for this "add the contents of a block into a free list" triple:

$$\begin{array}{l} \{B = \mathtt{align}(B, 2^S) * \mathsf{block}(B, E - B) * 1 \le S * L = \mathtt{null}\} \\ \mathtt{while}\ B + 2^S < E\ \mathtt{do} \\ \quad \big([B] := L; L := B; B := B + 2^S\big) \\ \{\mathsf{block}(B, E - B) * \mathtt{freelist}(L, S)\} \end{array}$$

[7 marks]

(*e*) Give a loop invariant, and precisely but informally explain why it is preserved, for this "coalesce blocks of a size S free list into a size $S + 1$ free list" triple:

$$\{\mathtt{freelist}(L1, S) * L2 = \mathtt{null} * D = 0\}$$

$$\mathtt{while}\ D = 0\ \mathtt{do} \left(\begin{array}{l} \mathtt{if}\ L1 = \mathtt{null}\ \mathtt{then}\ D := 1 \\ \mathtt{else} \left(\begin{array}{l} X := [L1]; \\ \mathtt{if}\ X = \mathtt{null}\ \mathtt{or}\ X\ \mathtt{mod}\ 2^{S+1} \neq 0\ \mathtt{then}\ D := 1 \\ \mathtt{else} \left(\begin{array}{l} Y := [X]; \\ \mathtt{if}\ X + 2^S \neq Y\ \mathtt{then}\ D := 1 \\ \mathtt{else}\ \big([X] := L2; L2 := X\big) \end{array} \right) \end{array} \right) \end{array} \right)$$

$$\{\mathtt{freelist}(L1, S) * \mathtt{freelist}(L2, S + 1)\}$$

[6 marks]