Consider a programming language that consists of commands $C$ composed from assignments $X := E$ (where $X$ is a program variable, and $E$ is an arithmetic expression), heap allocation $X := \text{alloc}(E_1, \ldots, E_n)$, heap assignment $[E_1] := E_2$, heap dereference $X := [E]$, disposal of heap locations $\text{dispose}(E)$, the no-op $\text{skip}$, sequencing $C_1; C_2$, conditionals $\textbf{if } B \textbf{ then } C_1 \textbf{ else } C_2$ (where $B$ is a boolean expression), and loops $\textbf{while } B \textbf{ do } C$. null is 0

(a) Explain informally what it means for a separation logic partial correctness triple $\{P\} C \{Q\}$ to be valid. [3 marks]

(b) Explain informally what it means in terms of the executions of $C$ for the separation logic partial correctness triple $\{\top\} C \{\bot\}$ to be valid. [2 marks]

(c) Recall the list representation predicate $\text{list}$:
\[
\text{list}(t, []) = (t = \text{null}) \quad \text{list}(t, h :: \alpha) = \exists y. ((t \mapsto h) \ast ((t + 1) \mapsto y) \ast \text{list}(y, \alpha))
\]
We write $[]$ for the empty mathematical list; $h :: \alpha$ for the mathematical list the head of which is $h$, and the tail of which is $\alpha$; $\alpha \ast \beta$ for the concatenation of mathematical lists $\alpha$ and $\beta$; $\alpha[i]$ for the $i$-th element of the list $\alpha$, starting at 0; and $[k, \ldots, n]$ for the ascending list of integers from $k$ to $n$, including $k$ and $n$.

Give a proof outline, including a loop invariant, for the following triple:
\[
\{N = n \land N \geq 0\} \quad X := \text{null}; \textbf{while } N > 0 \textbf{ do } (X := \text{alloc}(N, X); N := N - 1) \quad \{\text{list}(X, [1, \ldots, n])\}
\] [4 marks]

(d) Also recall the partial list representation predicate $\text{plist}$:
\[
\text{plist}(t, [], u) = (t = u) \\
\text{plist}(t, h :: \alpha, u) = \exists y. ((t \mapsto h) \ast ((t + 1) \mapsto y) \ast \text{plist}(y, \alpha, u))
\]

Give a loop invariant for the following list sum triple:
\[
\{\text{list}(X, \alpha)\} \\
Y := X; N := 0; \textbf{while } Y \neq \text{null} \textbf{ do } (M := [Y]; N := N + M; Y := [Y + 1]) \\
\{\text{list}(X, \alpha) \land N = \sum_{i=0}^{\text{length}(\alpha)-1} \alpha[i]\}
\] [4 marks]

(e) Give a loop invariant for the following list concatenation triple:
\[
\{\text{list}(X, \alpha) \ast \text{list}(Y, \beta)\} \\
\textbf{if } X = \text{null} \textbf{ then } Z := Y \textbf{ else } \\
Z := X; U := Z; V := [Z + 1]; \quad \textbf{while } V \neq \text{null} \textbf{ do } (U := V; V := [V + 1]); \\
[U + 1] := Y \\
\{\text{list}(Z, \alpha \ast \beta)\}
\] [5 marks]

(f) Describe precisely a stack and a heap that satisfy $\text{list}(X, [1, \ldots, 3])$. [2 marks]