Compiler Construction

A regular grammar is a grammar whose rules are in one of the two following forms (where $A$ and $B$ are non-terminal symbols and $a$ is a terminal):

$$
\begin{align*}
A & \rightarrow a \\
A & \rightarrow aB
\end{align*}
$$

(a) Give a regular grammar which generates floating point numbers of exactly the following form:

$$(0|1)^+.(0|1)^*[(e(0|1)^+)$$

where “()” indicates grouping, “[ ]” indicates optional item, “ρ+” indicates one or more repetitions of $\rho$ and “$\rho^*$” indicates zero or more repetitions of $\rho$. [8 marks]

(b) Give a non-regular grammar with fewer productions than your answer to (a) but which generates the same set of strings. [4 marks]

(c) Determine, with justification, for the following grammars

(i) whether $S$ generates strings not generated by $T$; and

(ii) whether $T$ generates strings not generated by $S$.

$$
\begin{align*}
S & \rightarrow aaS \\
S & \rightarrow Scc \\
S & \rightarrow d
\end{align*}
\quad \text{and} \quad
\begin{align*}
T & \rightarrow aTc \\
T & \rightarrow d
\end{align*}
$$

[4 marks]

(d) What is the significance for the compilation process of the idea of “strings which can be generated by regular grammars”? Your answer should explain where such a module recognising such strings would appear in a compiler and a possible external interface (functions, variables and/or objects) it might present to the rest of the compiler. [4 marks]