1 Concepts in Programming Languages (am21)

(a) Algol-60 provided two parameter-passing mechanisms: call-by-value and call-by-name.

(i) Explain these mechanisms. [2 marks]

(ii) Justify or criticise the statement that “the former is expensive for arrays and the latter interacts badly with side effects”. [2 marks]

(iii) What parameter-passing mechanism(s) do C and Java use, and how do such languages deal with an array being passed as a parameter? [2 marks]

(b) A side-effect-free call-by-value language has its ML-like syntax of expressions extended to be able to model call-by-name and (LISP-like) call-by-text:

\[
\begin{align*}
e & := \ldots \mid \text{suspend } e \mid \text{force } e \quad \text{(call-by-name)} \\
e & := \ldots \mid \text{quote } e \mid \text{eval } e \quad \text{(call-by-text)}
\end{align*}
\]

Both \text{suspend } e and \text{quote } e yield an unevaluated representation of \(e\) as a value for later evaluation by \text{force} and \text{eval} respectively. Sketch two programs (differing only in whether they use \text{suspend} and \text{force} or \text{quote} and \text{eval}) which give different results. [Note: Answers using side-effecting operators can only gain partial marks.]

[4 marks]

(c) A library defines a generic class \text{Foo<T>} in a Java-like language. A user’s program declares a class \(C\) and subclasses it as class \(D\), creating variables \(fc\) and \(fd\) of types \text{Foo<C>} and \text{Foo<D>} respectively.

(i) Construct a declaration of \text{Foo<T>} along with a program of the above form containing the assignment \(fc=fd\) which, if this statement were legal, would be the cause of a later run-time error when executed. [5 marks]

(ii) How might the language syntax be changed to optionally express that the above assignment is to be allowed, indicating any compensating restrictions required for the declaration of \text{Foo<T>} or \(fc\) to avoid run-time errors. [3 marks]

(iii) How do Java arrays of type \(T[]\) fit in with your answer to Part (c)(i)? [2 marks]