Optimising Compilers

(a) Explain two concepts of a variable being live – one related to execution behaviour and one related to the structure of a program. Relate them by implication, and explain their relative ease of computation in a compiler. [4 marks]

(b) Explain how live variable analysis can be used to allocate variables to registers by colouring. Give and justify an algorithm that performs this colouring, particularly noting how it avoids early decisions causing inconvenient early choices of colour. [5 marks]

Let $K_n$ be the graph of $n$ nodes, each having an edge to each other; let $C_n$ have $n$ nodes, but with $n$ edges arranged to give a cycle; and let $S$ be $C_4$ with an additional edge forming a diagonal of $C_4$ seen as a square.

(c) What is the minimum number of colours necessary to colour $K_n$, $C_n$ and $S$? [3 marks]

(d) How many colours does your algorithm require for $C_n$ (if it makes arbitrary choices give both best-case and worse-case)? [2 marks]

(e) Give programs that have $K_5$, $C_4$ and $S$ as colouring problems for register allocation. [3 marks]

(f) Give programs in SSA form (or indicate when this is impossible) for the three graphs in part (e). [3 marks]