Optimising Compilers

(a) Explain in one sentence what is meant by the “phase order problem”.  

[2 marks]

For the rest of the question, it is recommended that you restrict attention to a single basic block containing only unary and binary arithmetic instructions, e.g. add#4 dst,src or mul dst,src1,src2 and with no variable written to after being read (SSA form). Consider all such instructions to execute in one cycle.

(b) Describe how a directed acyclic graph expressing instruction dependencies suitable for instruction scheduling can be obtained from such a basic block.  

[4 marks]

(c) Briefly describe how a (register interference) graph can be obtained from such a basic block. Also state a requirement for colouring this graph with registers.  

[4 marks]

(d) Consider the two programs (where the first four lines are the same):

```
add t1,a,b
add#2 t2,c
add#3 t3,c
mul t4,t2,t3
sub z,t1,t4
```

```
add t1,a,b
add#2 t2,c
add#3 t3,c
mul t4,t2,t3
sub t5,t1,t4
xor t6,t5,a
xor z,t6,b
```

In each case consider scheduling the add t1,a,b to appear as early as possible or as late as possible. Determine the number of registers required to colour the program in all four resulting cases. (Assume that only z, allocated to r1, is live on exit and that a,b,c are allocated registers r1,r2,r3.)  

[10 marks]

[Remark: you might note that doing add t1,a,b in the left-hand program decreases the number of live registers by one, while it increases the number by one on the right-hand program.]