9 Optimising Compilers (tmj32)

The following function in C-style code is optimised by a compiler. Assume that variable arg0 is the argument to the function and so has already been defined.

```
x = arg0;
y = x * 2;
z = x * 4;
while (true) {
   if (x % 5 == 0) {
     y = z + 1;
     print(y);
     break;
   }
   x = x - 1;
}
y = arg0;
print(y);
```

(*a*) What is the live range of a variable? [2 marks]

(*b*) User variables are assumed to reside in the same virtual register in the intermediate representation throughout the entire program. How can static single assignment (SSA) form help reduce their live ranges? [2 marks]

(*c*) Put the code above into SSA form. [4 marks]

(*d*) Describe and give the dataflow equation for live variable analysis. [4 marks]

(*e*) Perform live variable analysis on the original code at the beginning of the question and use it to perform dead-code elimination, showing the *in-live* sets after the analysis. [4 marks]

(*f*) Describe how dataflow analyses, such as live variable analysis, could be simplified if the code was in SSA form. [4 marks]