Optimising Compilers 2012–2013
Exercise Sheet 2

The purpose of this exercise sheet is to practise *register allocation*, *strength reduction*, *static single assignment*, *abstract interpretation* and *strictness analysis*.

1. (a) Briefly describe the concept of *abstract interpretation*, making particular mention of safety.

   Consider a form of *abstract interpretation* in which our abstraction function captures the possible intervals of integer arithmetic. For example given a variable x the abstraction function α returns the interval $[l_x, h_x]$ where l_x is the lowest possible value of x and h_x is the highest possible value of x. For variables x and y the following properties hold for our abstraction function:

   $$f(x+y) = [l_x + l_y, h_x + h_y]$$
   $$f(x-y) = [l_x - h_y, h_x - l_y]$$

   (b) Given the following function calculate the interval of its return value in terms of the intervals of `x` and `y`.

```
int g(x, y) {
        int a = x-y;
        int b = x+x;
        return b+a;
}
```

   (c) An abstract interpretation of a program containing the function `g` ascertains the interval of the parameters to `g` as $\alpha(\texttt{x}) = [5, 10]$ and $\alpha(\texttt{y}) = [0, 2]$. Given this information can `g` return 0? Give the interval of `g`.

2. (a) Explain the notion of a 3-argument function being strict in its second parameter, considering both mathematical view of functions (an extra value $\perp$ representing non-termination, and the operational view of looping behaviour).

   (b) Do the functions $f(x, y) = f(x, y)$ and $g(x, y) = x + y$ have different strictness? Do they allow different strictness optimisations? Explain any differences between 'strict' in a parameter, and needing to evaluate it.

   (c) Give the strictness function for $f(x, y, z) = if\ x = 0\ then\ y\ else\ y + z$ and justify it.

   (d) Consider a weaker form of strictness analysis where the abstract value of an n-argument function is just an $n - argument$ bit vector where bit k is 1 if and only if the concrete function is strict in parameter k. Why is this weaker? Give a program for which strictness optimisation optimises a parameter to call-by-value but which this weaker analysis fails to optimise.

3. (a) Describe the purpose of register allocation and how graph colouring can help.

   (b) Describe a possible downside of a graph colouring approach in the context of JIT compilers.

   (c) Research an alternative register allocation algorithm (*hint: linear scan*) and briefly contrast it with the graph colouring approach.

Please also complete the following past exam questions:

- 2002 Paper 7 Question 4
- 2004 Paper 8 Question 7
- 2005 Paper 8 Question 7 (part (b))

Past exam questions can be found at:
`http://www.cl.cam.ac.uk/teaching/exams/pastpapers/t-OptimisingCompilers.html`