Software Design (F28SD2):
Exercises in Static Analysis

Andrew Ireland
School of Mathematical and Computer Sciences
Heriot-Watt University
Edinburgh

Exercise 1

Calculate the cyclomatic complexity for the following code.

```
boolean finddata(int val, int lower, int upper, int max, int data[]){
   int sum       = 0;
   int index     = lower;
   boolean found = false;

   if ((lower < 0) || (upper >= max))
      System.out.println("ERROR");
   else
      {while ((data[index] > 0) && (index <= upper) && !found)
          if (data[index] == val) found = true;
          else index++;
      }
   return found;
}
```

Note that compound conditions should be decomposed, *i.e.* each atomic condition should be treated as a distinct node when constructing the flow graph.

Exercise 2

Consider the following fragment of Java code:

```
int z = 0;
while (x > 0) {
   if odd(x) z = z+y;
   y = y*2;
   x = x div 2;}
screen.println("x = " + x);
screen.println("y = " + y);
```

Construct a backward program slice of the above code fragment with respect to the last occurrence of `x`

Exercise 3

Consider the following fragment of Java code:

```
int k = 1;
int s = 0;
while (k < 99) {
   s = s + k;
   k = k + 1;}
screen.println("s = " + s);
screen.println("k = " + k);
```

Construct a forward program slice of the above code fragment with respect to the first occurrence of `s`.