CS510'15 Assignment #1
(Due: Feb 03 in class)

January 22, 2015

1 CFG, Dominator, and Post-Dominator (25p)

1. Construct the Control-Flow Graph (CFG) for the code snippet below (10p).
2. Then, based on your CFG, construct the list of dominators and immediate post-dominators for statements 5, 7, 8, 9, 10, 11, and 14 (7p).

```
1  int n = input();
2  float k = 0f;
3  int j = 0;
4  float sum = 0;
5  if (n < 1)
6    return -1;
7  while (n > 0) {
8    if (n % 2 == 0) {
9      sum += -1 / (2*k-1);
10     for (j = 0; j < n; j++) {
11       sum += k/j;
12     }
13   } else {
14     sum += 1 / (2*k-1);
15   }
16   k = k + 1.0f;
17   n--;
18 }
```

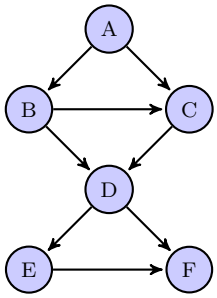
3. Prove that a statement has exactly one immediate post-dominator (8p).

2 Program Dependence Graph (20p)

Construct the PGD for the code in problem 1. If the graph becomes too crowded you may split it into two sub graphs: data dependence graph and control dependence graph.

3 Path Construction (10p)

1. Count the number of unique (i.e., without loops) complete paths through the CFG from problem 1 (5p).
2. Count the number of unique complete paths through the CFG below (5p):



4 Trace Compression (10p)

Let a plain text string be: *a b c d a a b c d e a a a b c d*. The initial lookup table is:

Context	Prediction
aa	a
bd	a
bc	d

.

Use FCM-2 to compress the string. The final compressed string and the final lookup table are required. Intermediate steps are not required but encouraged.

5 Path Profiling (25p)

```
if (p1) {
  s1;
} else {
  while (p2) {
    while (p3) {
      if (p4) {
        s2;
        continue;
      }
      if (p5)
        break;
    }
  }
  if (p6)
    s3;
}
```

1. Construct the path enumeration graph for the above program using the EPP algorithm. Show the path encoding.

2. Show the final instrumented program, including where instrumentation is inserted.

6 Predicate Tracing (20p)

Predicate tracing is a control flow tracing technique that records the branch outcomes of predicates. For example,

```
1 if (c1)
2   if (c2)
3     s0;
4   if (c3)
5     s1;
6 s2;
```

The trace 1 2 3 4 6 for the above program can be represented as T T F. Three bits are needed.

1. Please list the challenges for making the above idea work on real world programs. You can assume C or Java programming language semantics.

2. Sketch solutions to such challenges.

Using examples is encouraged.