Problem Set 3: White-Box Testing

1. Give a set of inputs for the method below which would achieve condition/decision but not modified condition/decision.

```java
int coverageMethod(int a, int b):
    if a > 10 or b < 25:
        return a + b
    else:
        return 0
```

2. How does the role of coverage metrics differ in black-box testing and white-box testing?

3. What is a basis path set?

4. What are two static checks we can perform using a data flow (def-use) graph?

5. What is the difference between a general control flow graph and a DD path graph? Why are both acceptable to use for analyzing path coverage? Which one is preferred and why?
6. In a few sentences, summarize how data flow testing and control flow (basis path) testing are similar and how they are different.

7. Draw a control flow graph for the Java method `controlFlowOne` given below.

```java
public int controlFlowOne( boolean a, int b ) {
    int c = 0;
    while( a ) {
        if( b > 0 ) {
            c = (c + b) * -1;
        }
        a = (c <= 0);
    }
    return c;
}
```

8. Generate a basis path set for `controlFlowOne`. You may use either the node labels from your control flow graph or the line numbers to represent your path.
9. Draw a data flow (def-use) graph for the Java method `dataFlowOne` given below.

```java
public String dataFlowOne ( int a, int b ) {
    String c = a + "-" + b;
    while ( c.length () < a + b ) {
        c = c + "-ITER";
    }
    return c;
}
```

10. List the elements of the sets below with respect to the method `dataFlowOne`, along with a subpath for each element. You may use either the node labels from your data flow graph or the line numbers to represent your path.

- `dcu(c, 2)`

- `dcu(c, 5)`

- `dpu(a, 1)`
11. Generate a set of test inputs that would realize all basis paths for `controlFlowOne`.

12. Generate a set of test inputs that would provide all-c-uses/some-p-uses coverage for the `dataFlowOne` method.