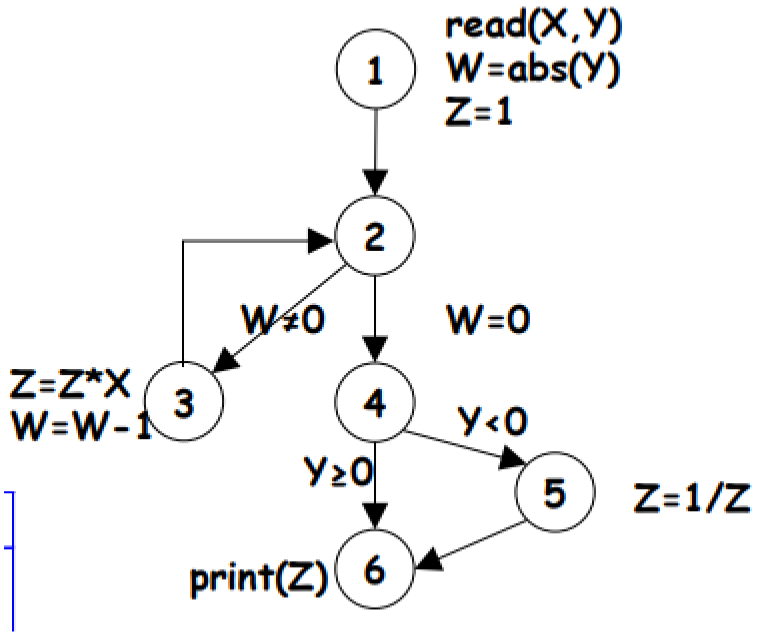
COEN 448 Software Testing and Validation Assignment 1 – White Box Testing

Individual assignment, due by March 8th 23:59.

In this assignment, we aim to practise white box testing and coverage strategies to produce unit test cases.

Problem 1 (30 Marks) : Given a CFG below, 1.1 please complete the table of def-use pairs.



- The nodes are {1, 2, 3, 4, 5, 6}. The program has variables {X, Y, Z, W};
- def(i) means at node i, the set of variables defined;
- c-use(i) is the set of variables that are c-used at node i;
- edge(i, j) means the set of directed connections from node i to node j;
- p-use(i, j) means the set of variables that are p-used at the edges (i,j);
- dcu(v,i) is the def-use pair that reads as variable v defined at node i. The pair is the set of node j that v is def-clear and c-used at node j.
- dpu(v,i) is the def-use pair that reads as variable v defined at node i. The pair is the set of edge (j, k) that v is def-clear and p-used at node j.

node i	def(i)	c-use(i)	edge(i,j)	p-use(i,j)

(10 Marks)

and

node i	dcu(v,i)	dpu(v,i)

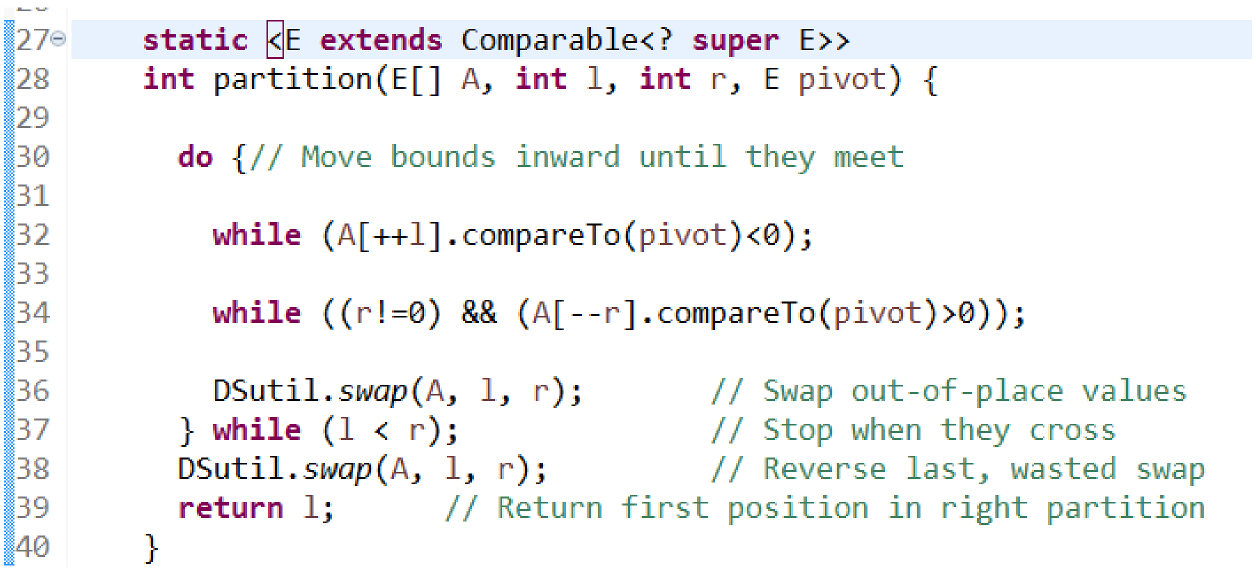
(10 Marks)

1.2 Given the dcu and dpu, write the test cases to cover them all. (10 Marks)

Problem 2 (60 Marks): The QuickSort algorithm code is provided in the attachment of this assignment. In QuickSort, the pivot is an element in the array. The function partition() returns the position of the pivot in the array so that the left side of the pivot has elements smaller to the value of the pivot while the right side of the pivot has the elements equal or bigger to the value of the pivot.

2.1 **(10 Marks) Develop test cases** following the black-box approach that has input domain modeling of the partition function according to the (2.1.a) best case, (2.1.b) worse case and (2.1.c) average case of the quick sort algorithm. You can choose base choice coverage or other coverage criterion to develop the test cases.

2.2 **(15 Marks) Write the unite test cases** for (2.1.a), (2.1.b) and (2.1.c) , run those test cases and produce the coverage report from your programming IDE.



```java
    static <E extends Comparable<? super E>>
    int partition(E[] A, int l, int r, E pivot) {

      do {// Move bounds inward until they meet

        while (A[++l].compareTo(pivot)<0);

        while ((r!=0) && (A[--r].compareTo(pivot)>0));

        DSutil.swap(A, l, r);       // Swap out-of-place values
      } while (l < r);              // Stop when they cross
      DSutil.swap(A, l, r);         // Reverse last, wasted swap
      return l;      // Return first position in right partition
    }
```

2.3 (20 Marks)(2.3.1)Produce a CFG of the partition function. Leverage the table based def-use pair approach in Question 1, and produce the table below for variable *pivot*.

node i	dcu(v,i)	dpu(v,i)

(2.3.2) Develop test cases to cover all the dcu and dpu.

(2.3.3) Program unit test cases, run the test cases and produce a coverage report from you IDE.

Submission:

1. A report document that writes the following section.
    - **Solutions** to Problem 1 and Problem 2 including tables, test cases, CFG and screenshots of the coverage reports.
    - Observe the coverage reports of Problem 2. **Discussion** of the difference in the coverage and explain the possible cause of the difference.
    - **Comments** on the pros and cons of the two testing approaches (input domain modeling and data flow testing).
2. The unit test source code.

Please pack all the submission in one archive file with .zip or .tar. No .rar files are acceptable.