Prof. Dr.-Ing. Peter Liggesmeyer
M.Sc. Rasha Abu Qasem
M.Sc. Nishanth Laxman

TU Kaiserslautern
Dept. of Computer Science
Lehrstuhl Software Engineering: Dependability

Software Quality Assurance (WS20/21)

Problem Set 3

Problem 1: Data Flow Oriented Test

Given is the function `sum`:

```
public static int sum(int n) {
 int sum = 0;
 int i;
 for (i = 1; i <= n; i++) {
   sum = sum + i;
 }
 return sum;
}
```

a) Please create a control flow diagram with data flow annotation for the function `sum`.
b) Write down all def-use pairs in a table as in the example below. Indicate p-uses and c-uses.
c) Please determine the minimal necessary test path for fulfilling the ***all defs*** criterion of the `sum` function. Please denote the required test path and mark this path in the control flow diagram.
d) Please determine the minimal necessary test path for fulfilling the ***all c-uses*** criterion for the `sum` function. Please denote the required test path and mark this path in the control flow diagram.
e) Please determine the minimal necessary test path for fulfilling the ***all p-uses*** criterion for the `sum` function. Please denote the required test path and mark this path in the control flow diagram.
f) Please determine the minimal necessary test path for fulfilling the ***all c-uses/some p-uses*** criterion for the `sum` function. Please denote the required test path and mark this path in the control flow diagram.

Hints:
- the path to a c-use ends in the node of the use
- p-uses have two paths (one for each path of the decision)

Example:

Use	Defined in	Path	Variable
p1	n1	n1,n2,n4	x
p2	n1	n1,n2,n3	x
c1	n1	n1,n2,n3,n5	y

Problem 2: Path Coverage Test

a) Please determine the minimal necessary test cases for fulfilling the structured path coverage test for the parameter ***k=1*** for the `sum` operation.

b) Please determine the minimal necessary test cases for fulfilling the boundary interior test for the sum operation.

Problem 3: Data flow oriented test

Given is a section of code for sorting a one-dimensional integer field using the bubble sort algorithm. The corresponding control flow diagram is presented as well.

```
public static int[] elements = {42,4,8,23,15,16};
public static int length() {return elements.length;}
public static int get(int i) {return elements[i];}
public static void put(int i,int x) {elements[i]=x;}
public static void bubblesort() {
  int a0,a1,j;
  int i=length()-1;
  while (i>=0) {
    j=0;
    while (j<i) {
      a0=get(j);
      a1=get(j+1);
      if (a0>a1) {
        put(j,a1);
        put(j+1,a0);
      }
      j++;
    }
    i--;
  }
}
```

a) Draw the control flow diagram of the function bubblesort(), with the missing data flow attributes for a data flow oriented test.Give the minimal necessary test path for the fulfillment of statement coverage. Briefly explain the definition of the statement coverage criterion.

b) Give the minimal necessary test path for the fulfillment of branch coverage. Briefly explain the definition of the branch coverage criterion.

c) Fill out the following table with all def-p-use pairs P.

P	Def	Path	Variable
p1			
p2			
p3			
p4			
p5			
p6			
p7			
p8			
p9			
p10			
p11			
p12			
p13			
p14			
p15			
p16			
p17			
p18			

d) Give the minimal necessary *def-use-pairs* for the fulfillment of the *all-p-uses/some-c-uses* test. Utilize the set of *def-p-use-pairs* identifiers in the table, e.g. {p1, p2}. Briefly state the definition of this criterion.

Problem 4: State-based Test

Given is the specification of a digital watch software.
For adjustment of a digital watch, the following states are to be considered:

- *Normal time:* State after inserting the battery
- *Adjust Hours:* Hours can be adjusted
- *Adjust Minutes:* Minutes can be adjusted
- *Adjust Seconds:* Seconds can be adjusted

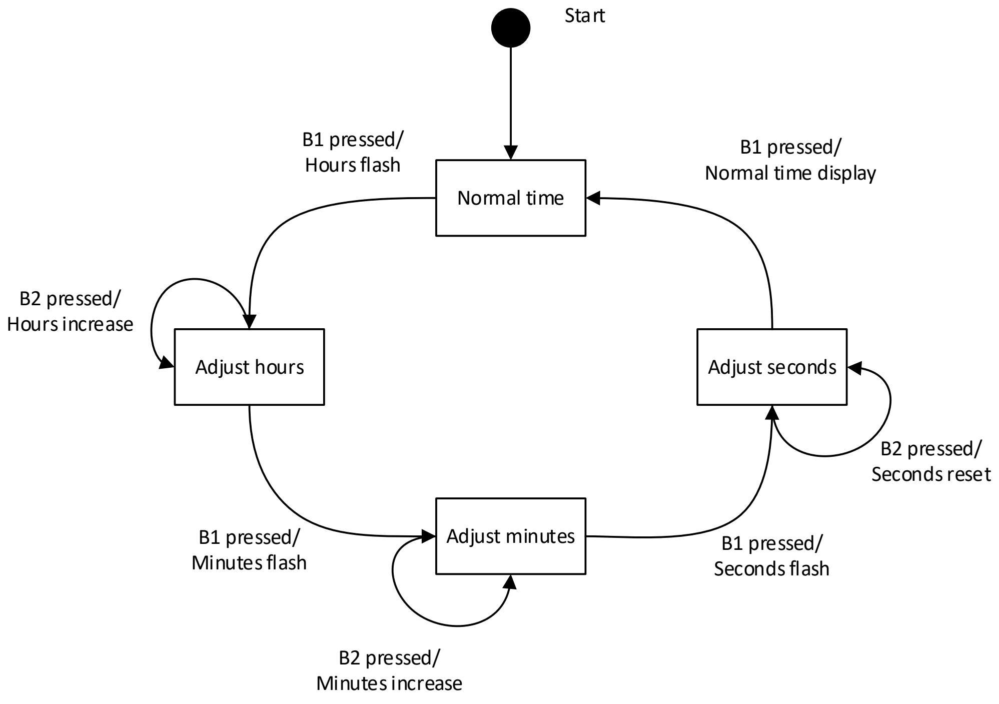
The following events could occur:

- *Start signal:* Battery inserted
- *Button 1 pressed*
- *Button 2 pressed*
- The two buttons must not be pressed simultaneously.

The following outputs could happen:

• *Hours flash:*	The operator is currently in the hour editing mode.
• *Minutes flash:*	The operator is currently in the minute editing mode.
• *Seconds flash:*	The operator is currently in the second editing mode.
• *Hours increase:*	The hour display has increased by 1 hour.
• *Minutes increase:*	The minutes display increases by 1 minute.
• *Seconds reset:*	00 displays as second display.
• *Initialization:*	Display of 00:00:00

State chart “Watch adjustment“


Start
B1 pressed/ Hours flash
Normal time
B1 pressed/ Normal time display
B2 pressed/ Hours increase
Adjust hours
Adjust seconds
B2 pressed/ Seconds reset
B1 pressed/ Minutes flash
Adjust minutes
B1 pressed/ Seconds flash
B2 pressed/ Minutes increase

a) Please determine the test data for the program execution that traverses every state. Please select the simplest test cases.
b) Please determine the test data for the program execution that traverses every transition. Please select the simplest test cases.