TESTING LOOPS

CSC Software Testing Materials (Testing: Loops)

@ Dr. Jessica Young Schmidt and NCSU Computer Science Faculty

TESTING REVIEW

- System Testing
- Unit and Integration Testing

TESTING LOOPS

Pressman provides the following guidance for testing a simple loop (i.e., no nesting), where the loop is expected to iterate n times.

- Fail the conditional test for entering the loop, so that the loop never executes;
- Execute the body of the loop only once;
- Execute the body of the loop twice;
- Execute the body of the loop m times, where m < n ;
- Execute the body of the loop n – 1 times;
- Execute the body of the loop n times; and
- Execute the body of the loop n + 1 times.

Pressman, R. S. (2005). Software Engineering: A Practitioner's Approach(6th ed.). McGraw-Hill.

UPDATED REQUIREMENTS FOR PAYCHECKS

- Loops to allow for processing more than one paycheck at a time
- Extensive error checking

Multiple Paychecks

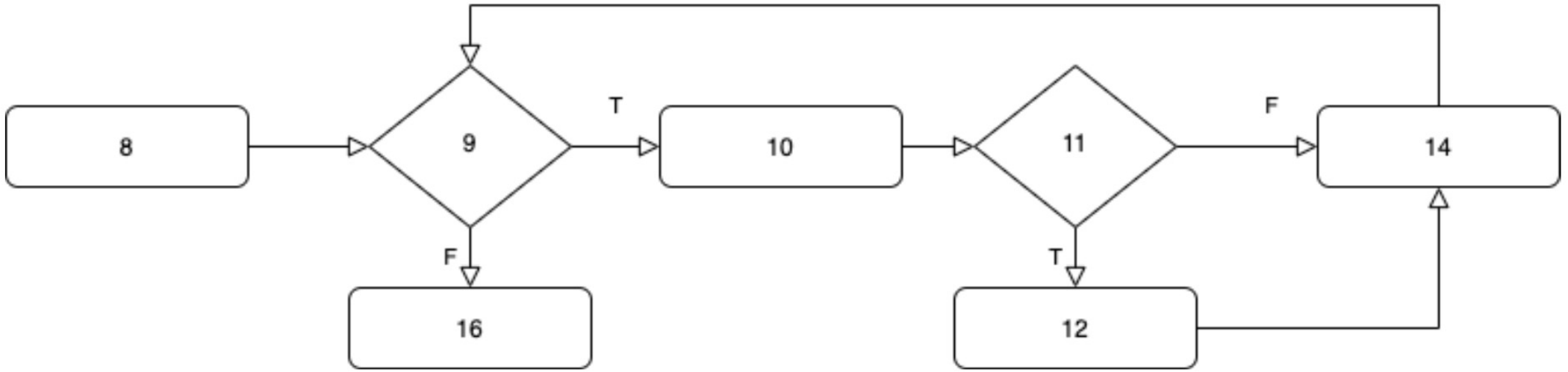
```
$ java -cp bin RobustPaycheck
New paycheck (Y/N):
N

$ java -cp bin RobustPaycheck
New paycheck (Y/N):
Y
Employee Name: Ellen Edwards
Employee Level: 1
Hours Worked: 39
Medical Insurance (Y/N): N
Dental Insurance (Y/N): N
Vision Insurance (Y/N): Y
New paycheck (Y/N):
N
```

```
$ java -cp bin RobustPaycheck
New paycheck (Y/N):
Y
Employee Name: George George
Employee Level: 3
Hours Worked: 41
Medical Insurance (Y/N): N
Dental Insurance (Y/N): Y
Vision Insurance (Y/N): N
Retirement Percentage (0-6): 2
New paycheck (Y/N):
Y
Employee Name: Hilda Henderson
Employee Level: 3
Hours Worked: 50
Medical Insurance (Y/N): N
Dental Insurance (Y/N): N
Vision Insurance (Y/N): N
Retirement Percentage (0-6): 3
New paycheck (Y/N):
N
```

getHoursWorked

```
1   /**
2    * Returns the hours worked for a given employee as entered by the user.
3    *
4    * @param console Scanner for reading from the console
5    * @return hours worked
6    */
7   public static double getHoursWorked(Scanner console) {
8     double hoursWorked = 0;
9     while (hoursWorked <= 0) {
10      System.out.print("Hours Worked: ");
11      if (console.hasNextDouble()) {
12        hoursWorked = console.nextDouble();
13      }
14      clearLine(console);
15    }
16    return hoursWorked;
17  }
```



getHoursWorked

```
    /**
     * Test the RobustPaycheck.getHoursWorked() method.
     */
    @Test
    public void testGetHoursWorked() {
        // Test boundary value
        Scanner input = new Scanner("1\n");
        assertEquals(1, RobustPaycheck.getHoursWorked(input), DELTA,
                "Test boundary value for get hours worked");

        // Test boundary value
        input = new Scanner("0.1\n");
        assertEquals(0.1, RobustPaycheck.getHoursWorked(input), DELTA,
                "Test boundary value for get hours worked");

        // Test mid regular value
        input = new Scanner("20.5\n");
        assertEquals(20.5, RobustPaycheck.getHoursWorked(input), DELTA,
                "Test mid regular value for get hours worked");

        // Test invalid first value
        input = new Scanner("0\n 60\n");
        assertEquals(60, RobustPaycheck.getHoursWorked(input), DELTA,
                "Test invalid first value for get hours worked");

        // Test invalid first value
        input = new Scanner("number 3\n 60\n");
        assertEquals(60, RobustPaycheck.getHoursWorked(input), DELTA,
                "Test invalid first value for get hours worked");
    }
```