

# Java Programming and Testing

## Part 1

*by* Rhys Harding

---

|                |                                                                               |                 |       |
|----------------|-------------------------------------------------------------------------------|-----------------|-------|
| FILE           | 12016_RHYS_HARDING_JAVA_PROGRAMMING_AND_TESTING_PART_1_159704_1746385804.DOCX |                 |       |
| TIME SUBMITTED | 18-JAN-2019 11:28PM (UTC+0000)                                                | WORD COUNT      | 4044  |
| SUBMISSION ID  | 99458695                                                                      | CHARACTER COUNT | 23724 |

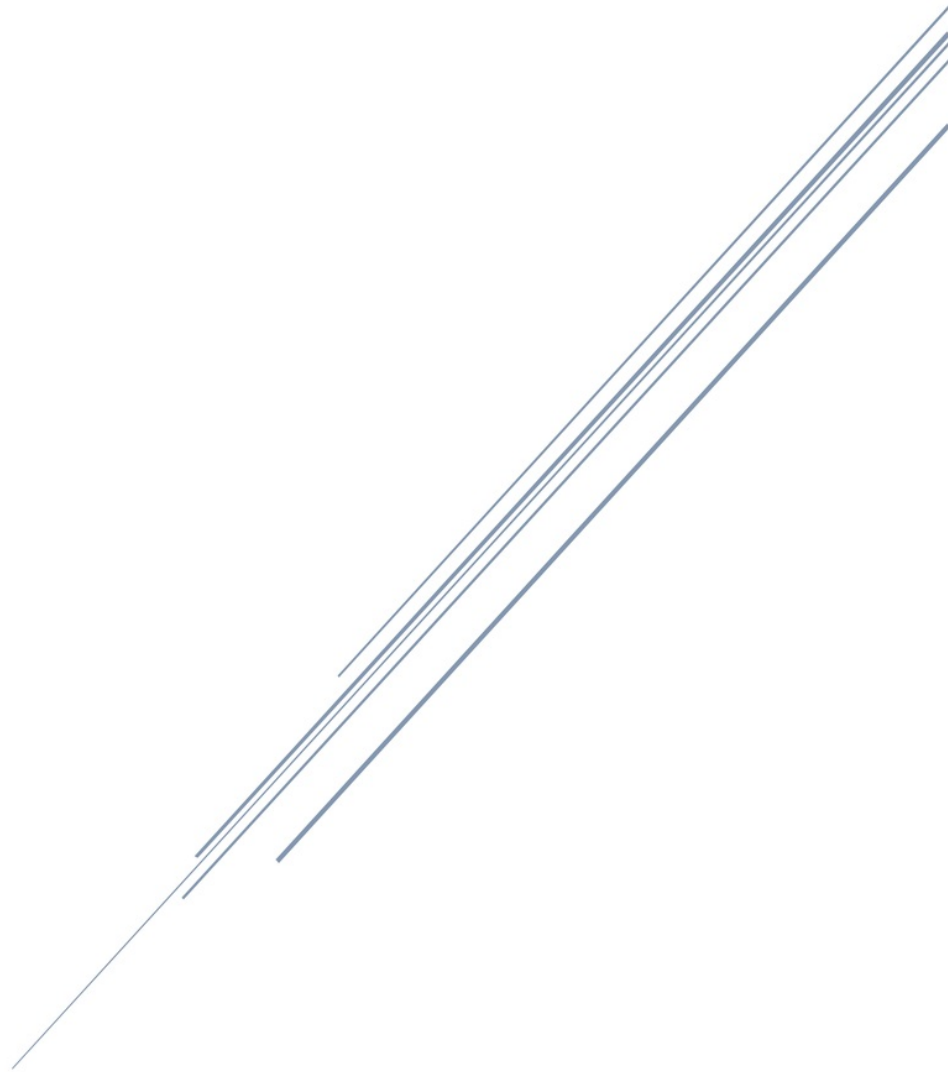
---

*MSC Software Development*

---

# PROGRAMMING AND SOFTWARE TESTING

Java Programming and Testing Part 1 a – Testing at the DVLA



**Rhys Harding**  
1809709

## Contents

|                                        |          |
|----------------------------------------|----------|
| <b>INTRODUCTION .....</b>              | <b>2</b> |
| <b>AGILE PROCESS .....</b>             | <b>2</b> |
| TESTING AT THE LOCAL ENVIRONMENT ..... | 2        |
| TESTING AT THE INTEGRATION LAYER ..... | 3        |
| <i>Performance Testing</i> .....       | 4        |
| • Load Testing .....                   | 4        |
| • Stress Testing .....                 | 4        |
| • Soak Testing .....                   | 4        |
| <b>CONCLUSION .....</b>                | <b>4</b> |
| <b>REFERENCES .....</b>                | <b>5</b> |

## Introduction

The Driver and Vehicle Licencing Agency (DVLA) carry out extensive testing of their software products at various phases. These testing phases differ in approach and each are executed with the intention to check that the product in hand is on course for completion. Moreover, whether the software contains any faults that need to be attended to. This report will outline the software testing techniques deployed at the DVLA based on current research, the authors observations and initial training; to which agile principles will be applied. Furthermore, an explanation of each testing technique will be presented.

## Agile Process

At the DVLA, agile software development practices are thoroughly implemented through the various tools that closely follow agile frameworks [1], [2] (more on this framework can be found in the Agile Manifesto of 2001) [3], [4]. The tools utilised, such as Scrum [5], [6], [7], Kanban [7] [8], [9] and Lean [7], [8] are iterative and incremental [10]. Therefore, software testing is an important aspect of the Software Development Life Cycle (SDLC) [11], [12], [13]. The DVLA takes its products from requirement analysis through to a sequence of phases [5], [14]. These phases incorporate testing at different stages, each utilising a different approach to testing [15]. What is more is that each stage holds value at different levels of importance, from cost at which the tests come at, the amount of testing required for different approaches, the time taken to carry out each phase of testing and whether they lay closer in perspective to the user or the developer [16] [14].

The importance of testing thoroughly at each stage, is a feature of the agile approach where testing occurs right from the earlier stages of development [2] [13], [15], [17], [15]. From a financial point of view, it is increasingly more expensive or unnecessary to correct a fault within code that has been detected much further down the development line [2], [15]. The quicker and earlier a team of developers and testers can identify a problem, the more time available to correct it and more money they save in the long run. Not only this, perhaps even more so, is that often in the earlier stages of development, when the software is in its infant period, the code is a lot easier to correct without causing much disruption [13]. More importantly, once a correction is made, the software once again must go through the testing phases, subsequently costing more time and money. The iterative nature of agile handles this possibility of constant change and rightfully insists upon the vigorous re-testing of the software [14], [16], [15]. It is obvious to associate testing with faults in code, however, testing is a check that the software product is meeting the requirements that were set out for that particular increment in the project [11], [13]. For such a software product, agile frameworks make is abundantly clear that testing ensures that the development is kept on track, specifically meeting each requirement which results in a high-quality product [2], [12], [18].

## Testing at the Local environment

At the time of writing code within a local branch of a particular repository, through a version control system, the developer is also tasked with writing unit test and acceptance tests and will do so on the local machine environment [11]. Unit testing is making sure a function does what is expected [11]. It is where the very primitive parts of the code are tested. Often, these

unit tests are written before the feature has even been coded and in these cases Test-Driven Development Practices are used [19], [20].

Test-Driven Development is a philosophy in which the requirements for a particular feature are broken down into explicit unit tests [19], [20]. These tests are designed to fail to begin with but the developer (and there are usually two or more people who operate here) [11], [20] start to develop the code to pass these tests and after every iteration, developing code that pass a corresponding series of tests [19]. The idea here being that the requirements take precedent and dictate the path of developing code so that once the code finally passes the tests in the most efficient way, the requirements are sufficiently fulfilled.

Within this environment the team must also run manual testing, exploratory testing and also accessibility testing [11]. Manually testing will involve the tester manually going through the test cases looking for bugs and defects [21]. However, manual testing can be time consuming and uses more human resources compared to automated testing, it offers the opportunity for the developers themselves to observe the tests, which is preferable when customer experience is a priority [21]. Exploratory testing refers to the approach the tester uses and the way in which they structure their time and resources [11]. The tester must improve their work, and alongside testing, must generate new tests to run, improving through experience and personal freedom. For those individuals with specific disabilities (including colour-blindness, hard of hearing and old age) use of the web is far more challenging [22] [23], [24], [25]. Accessibility testing is a subset of usability testing, testing usability to those with such disabilities [22].

### Testing at the Integration Layer

There also exists an environment called the Integration layer [11], where a Smoke test [26] is performed. The Smoke test is a very minimal test to check whether something is 'alive', and from the specific point of view of an end user, if the system is doing something [26]. After every commit a continuous integration tool, Drone is utilised [27]. Drone is integrated as part of the version control system that is Git, and will detect a commit and as a minimum, will run a feature test [11]. This ensures that the new development integrates with the feature and only when completed, will a pull request be issued.

Then the feature is reviewed by a tester and if satisfied the feature is merged [20]. It is at this point where the feature tests are re-run and a regression package is used in another development environment which is Amazon Web Services (AWS) [28]. This regression test collates the feature tests and checks to see whether the merge of old and new code introduced errors that were not previously present. This process of examining the joins of introduced software is called Component Integration Testing and the teams at DVLA use Cucumber [29] to write these tests [11]. It is important to test these points of integration to not only to check each feature, but also whether the new code that is introduced performs as it did before merger, and whether the code it is being merged with performs all the same [13], [20].



Integration tests can be tests of software on software or software on hardware and test the connections between systems, rather than what is in the system [13]. The author has observed that it is at this point all the tests are run swiftly. These tests need to be quick since

the faster the team receive feedback the cheaper it will be to put right [11]. The team make frequent commits to better identify and rectify errors as soon as they are made or at least as soon as possible to avoid extra costs [2]. At this point, the cyber security team will take a part in testing and perhaps also the business may want to implement some user testing [11]. These tests take place in the User Experience Lab that the DVLA have [11] and its purpose is to have actual users to use the product while the developers observe any points of friction, and identify any issues, again, at an early stage. The team will usually instruct the users to perform the various tasks of the software so that any usability infringements can be noted [8].

### Performance Testing

During the integration layer the team deploy certain measures of formal performance testing. These include:

- *Load Testing* – This type of testing consists of loading the software up to its limits of performance whilst observing the various metrics that come from the system under pressure [26], [30]. The author observed that the teams perform this regularly.
- *Stress Testing* – This involves extending the limits of the system [26], to deliberately fail [13], [30]. This might seem counterintuitive; however, it is useful for the team to observe *how* exactly the system failed. The result may be that a defect may appear that may otherwise go unnoticed. More often than not, especially for a live system, it is not possible to degrade the whole system into failing, so only one component is made to fail at a time. Although even this is difficult to do, the system is able to run in the meantime [11].
- *Soak Testing* – Soak testing is done by loading the system and running it over a long period of time [26]. This can give an indication to possible problems such as memory leaks and although the team's use of spring-framework makes this a little easier this problem can potentially still happen [11].

### Conclusion

The DVLA value software testing at the highest level, whilst agile tools help incorporate testing into every element of the development process [2], [11], [15], [15]. Not only is it the job of the testers to test but the developers must also be aware of the testing procedures that range from unit and acceptance tests to integration and performance tests. Testing is implemented at the very beginnings of a project, even before the software they are written for has been coded. They help the team to design to accomplish requirements and help the team spot flaws and defects in the code whilst under a host of different scenarios. The DVLA teams have tests at multiple stages and within multiple environments. This is due to the importance of continuous testing and ensuring every iteration implements clean working code.

## References

- [1] GOV.UK Service Manual , "Agile delivery. Agile methods: an introduction," Agile delivery community, 2016. [Online]. Available: <https://www.gov.uk/service-manual/agile-delivery/agile-methodologies>. [Accessed 18 December 2018].
- [2] Gov.UK Service Manual, "Agile delivery, Agile and government services: an introduction," Agile delivery community, 2016. [Online]. Available: <https://www.gov.uk/service-manual/agile-delivery/agile-government-services-introduction>. [Accessed 3 January 2019].
- [3] M. Beedle, A. van Bennekum, A. Cockburn, W. Cunningham, M. Fowler, J. Highsmith, A. Hunt, R. Jeffries, J. Kern, B. Marick, R. C. Martin, K. Schwaber, J. Sutherland and D. Thomas, "Manifesto for Agile Software Development," 2001. [Online]. Available: <http://agilemanifesto.org/>. [Accessed 30 November 2018].
- [4] M. Fowler, "Writing The Agile Manifesto," 2006. [Online]. Available: <https://martinfowler.com/articles/agileStory.html>. [Accessed 18 December 2018].
- [5] A. Powell-Morse, "Scrum – What is it and how do you use it?," Airbrake, 27 March 2017. [Online]. Available: <https://airbrake.io/blog/sdlc/scrum-what-is-it-and-how-do-you-use-it>. [Accessed 19 December 2018].
- [6] J. Sutherland , Scrum: The Art of Doing Twice the Work in Half the Time, Crown Publishing Group, 2015.
- [7] T. Dingsøyr, S. Nerur, V. Balijep and N. B. Moe, "A decade of agile methodologies: Towards explaining agile software development," *Journal of Systems and Software*, vol. 85, no. 6, pp. 1213 -1221, 2012.
- [8] Gov.UK Service Manual, "Agile delivery, Agile tools and techniques," Agile delivery community, 2016. [Online]. Available: <https://www.gov.uk/service-manual/agile-delivery/agile-tools-techniques>. [Accessed 22 December 2018].
- [9] A. Powell-Morse, "Kanban Methodology: What is it and how do you use it?," Airbrake, 5 April 2017. [Online]. Available: <https://airbrake.io/blog/sdlc/kanban>. [Accessed 2 January 2019].
- [10] C. Larman ; V.R. Basili, "Iterative and incremental developments. a brief history," *Computer*, vol. 36, no. 6, pp. 47 - 56, 2003.
- [11] N. Brooks-Thomas , *Software Testing at the DVLA*, Swansea: DVLA, 2018.
- [12] Gov.Uk Servive Manual, "Agile delivery: Planning in agile," 2016. [Online]. Available: <https://www.gov.uk/service-manual/agile-delivery/planning-agile>. [Accessed 20 December 2018].
- [13] M. E. Khan and F. Khan,, " Importance of Software Testing in Software Development Life Cycle," *International Journal of Computer Science Issues (IJCSI)*, vol. 11, no. 2, pp. 120-123,, 2014.
- [14] Gov.UK. , "Service Manual: Agile Delivery, Core principles of agile," Agile delivery community, 2016. [Online]. Available: <https://www.gov.uk/service-manual/agile-delivery/core-principles-agile>. [Accessed 2 January 2018].
- [15] S. Misra, "Agile Software Development Practices: Evolution, Principles, and Criticisms," *The International Journal of Quality & Reliability Management*, vol. 29, no. 9, p. 972–980, 2012.

- [16] G. Minton, "Revisiting Proven Development Processes," *Mortgage Banking Association of America*, vol. 71, no. 12, pp. 88-89, 2011.
- [17] S. M. Biju, "Agile software development methods and its advantages," in *Technological Developments in Networking, Education and Automation*, K. Elleithy, T. Sobh, M. Iskander, M. A. Karim, K. A. Mohammad and V. Kapila, Eds., Netherlands, Springer, 2010, p. pp. 603–607..
- [18] SINGH, A. and KAUR, P.J.,, "A SIMULATION MODEL FOR INCREMENTAL SOFTWARE DEVELOPMENT LIFE CYCLE MODEL," *International Journal of Advanced Research in Computer Science*, vol. 8, no. 7, 2017..
- [19] M. Soeken, R. Wille and R. Drechsler, "Assisted Behaviour Driven Development Using Natural Language Processing," in *50th International Conference on Objects, Models, Components, Patterns, TOOLS*, Prague, 2012.
- [20] S. W. Ambler, "Agile Testing and Quality Strategies: Discipline Over Rhetoric," 2013. [Online]. Available: <http://www.ambysoft.com/essays/agileTesting.html>. [Accessed 18 December 2018].
- [21] SALEEM, R.M., QADRI, S., UL HASSAN, I., BASHIR, R.N. and GHAFOR, Y.,, "Testing Automation in Agile Software Development 9(2), pp. 541-546.," *International Journal of Innovation and Applied Studies*,, vol. 9, no. 2, pp. 541 -546, 2014.
- [22] Gov.UK, Service Manual:, "Testing for accessibility," 2018. [Online]. Available: <https://www.gov.uk/service-manual/technology/testing-for-accessibility>. [Accessed 6 January 2018].
- [23] A. Arch, "Web Accessibility for Older Users: A Literature Review," 2008. [Online]. Available: <https://www.w3.org/TR/wai-age-literature/>. [Accessed 19 December 2018].
- [24] British Dyslexia Association, "A small victory - web accessibility," 2016. [Online]. Available: <https://www.bdadyslexia.org.uk/news/item/name/a-small-victory-web-accessibility>. [Accessed 19 December 2018].
- [25] British Dyslexia Association, "EU Web Accessibility Directive comes into effect in the UK bringing online learning accessibility changes," 2018. [Online]. Available: <https://www.bdadyslexia.org.uk/news/item/name/eu-web-accessibility-directive-comes-into-effect-in-the-uk-bringing-online-learning-accessibility-changes>. [Accessed 3 January 2018].
- [26] R. Khan and M. Amjad, "Smoke Testing of Web Application Based on ALM Tool," in *Computing, Communication and Automation (ICCCA), 2016 International Conference*, , 2016.
- [27] Drone.IO, Inc, "Drone," 2015. [Online]. Available: <https://drone.io/>. [Accessed 19 December 2018].
- [28] Amazon Web Services, Inc., "Start Building on AWS Today," 2019. [Online]. Available: <https://aws.amazon.com/>. [Accessed 10 January 2019].
- [29] Cucumber, "Cucumber: Tools & techniques that elevate teams to greatness," n.d. [Online]. Available: <https://cucumber.io/>. [Accessed 6 January 2019].
- [30] A. Shen ; M. Kuzlu ; M. Pipattanasomporn ; S. Rahman ; L. Chen, "A performance testing method for embedded software platforms," in *2016 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*, 2016.

[31] S. W. Ambler, "Agile Testing startegies," 2013. [Online]. Available: <http://www.ambyssoft.com/essays/agileTesting.html>. [Accessed 3 January 2019].

## Two Sets Coding

```
1.  /*
2.   * To change this license header, choose License Headers in Project Properties.
3.   * To change this template file, choose Tools | Templates
4.   * and open the template in the editor.
5.   */
6.  package twoarrays;
7.  import java.util.Scanner;
8.  /**
9.   *
10.   * @author Rhys
11.   */
12. public class TwoArrays {
13.     static Scanner stdin = new Scanner(System.in);
14.     /**
15.      * @param args the command line arguments
16.      */
17.     public static void main(String[] args) {
18.         // Rules
19.         System.out.println("This application will take two sets of any size (set A and set B), and \n"
20.             + "compare both using a variety of different techniques. There must not be\n"
21.             + "any duplicaates within each set. Navigate through the menu using the \n"
22.             + "corresponding numbers and execute each option as you like. When \n"
23.             + "finished, use the quit option to end the application. In the case where\n"
24.             + "the intersection or set difference appears to be empty, '[]' denotes \n"
25.             + "the 'empty set', the set containing no elements.");
26.
27.         layoutBreak(); // breaks up the display on the console
28.
29.         int[] setA = readInputArray("Input the values of set A: ", "Input the size of the first set, set A:
30. ");
31.         layoutBreak();
32.
33.         int[] setB = readInputArray("Input the values of set B: ", "Input the size of the second set, set B
34. : ");
35.
36.         System.out.println("-----Sets Recorded-----");
37.
38.         int choice; //
39.         do{
40.             System.out.print("Please select which function you wish to perform:\n"+
41.                 "
42.                 1. Display the numbers in both sets\n" +
43.                 "
44.                 2. Determining the maximum and the minimum values of both sets\n" +
45.                 "
46.                 3. Calculate the average of the numbers in each set\n" +
47.                 "
48.                 4. Determine if sets A and B overlap\n" +
49.                 "
50.                 5. If possible, determine the intersection of the A and B\n" +
51.                 "
52.                 6. If possible, determine the set difference A - B\n" +
53.                 "
54.                 7. Quit\n");
55.             choice = readInputInteger("Selection: ");
56.             switch (choice){ // menu
57.                 case 1:
58.                     printArray(setA);
59.                     printArray(setB);
60.                     break;
61.                 case 2:
```

```

55.         int[] minAndMaxA = findMinAndMax(setA);
56.         int[] minAndMaxB = findMinAndMax(setB);
57.         System.out.printf("For set A, the minimum value is %d, and the maximum value is %d.\n"

58.             + "For set B, the minimum value is %d, and the maximum value is %d.\n",
59.             minAndMaxA[0],
60.             minAndMaxA[1],
61.             minAndMaxB[0],
62.             minAndMaxB[1]);
63.         break;
64.     case 3:
65.         double averageA = findAverage(setA);
66.         double averageB = findAverage(setB);
67.         System.out.printf("The average for set A is %.2f, and the average for set B is %.2f.\n"

68.             ,
69.             averageA,
70.             averageB);
71.         break;
72.     case 4:
73.         if (doSetsOverlap(setA, setB)){
74.             System.out.println("Sets overlap.");
75.         } else {
76.             System.out.println("Sets do not overlap.");
77.         }
78.         break;
79.     case 5:
80.         if (intersection(setA, setB).length==0){
81.             System.out.println("The set 'A intersection B' is: [0]");
82.         } else {
83.             System.out.print("The set 'A intersection B' is: ");
84.             printArray(intersection(setA, setB));
85.         }
86.         break;
87.     case 6:
88.         if (setDifference(setA, setB).length==0){
89.             System.out.println("The set difference 'A - B' is: [0]");
90.         } else {
91.             System.out.println("The set difference 'A - B' is: ");
92.             printArray(setDifference(setA, setB));
93.         }
94.         break;
95.     case 7:
96.         System.out.println("Application closing.");
97.         break;
98.     default:
99.         System.out.println("Invalid selection.");
100.        break;
101.    }
102.    layoutBreak();
103.    } while (choice!=7);
104. }
105. /**
106.  * A method that reads the input from the user after displaying a prompt
107.  * @param message prompt
108.  * @return user input
109.  */
110. private static int readInputInteger(String message){
111.     boolean validInput = false; //validation to ensure the user enters an integer
112.     int integer = 0;
113.     while (!validInput){
114.         System.out.print(message);
115.         try {
116.             integer = Integer.parseInt(stdin.nextLine());
117.             validInput = true;
118.         } catch (NumberFormatException e){
119.             System.out.println("Invalid input! Please input an integer using the number keys.");
120.         }
121.     }

```

```

121.         return integer;
122.     }
123.     /**
124.      * A method that will read the user input into an array
125.      * @param prompt prompts the user to enter a value
126.      * @param promptSize prompts the user to specify an array size
127.      * @return the array that the user inputs
128.      */
129.     public static int[] readInputArray(String prompt, String promptSize){
130.         int arraySize = readInputInteger(promptSize);
131.         int[] array = new int[arraySize];
132.         int holder; // a variable that is intended to temporarily store the user input whilst duplicate val
            idation is being done
133.         boolean found = false;
134.         System.out.println(prompt);
135.         for (int i=0; i<array.length; i++){
136.             holder = readInputInteger("Value: ");
137.             for (int j=0; j<i; j++){ //reads back through the array that has already been written
138.                 //System.out.println(array[j]+" is j with index" + i);
139.                 if (holder==array[j]){ //duplicate
140.                     found = true;
141.                 }
142.             }
143.             if (found) {
144.                 System.out.println("Duplicate! Please enter a unique number.");
145.                 i--; //decrements in order to re-write in case of a duplicate
146.             } else {
147.                 array[i] = holder; //finally once all validation is done, the array takes on the value of the h
            older
148.             }
149.             found = false; //resets the found boolean ready for the next input
150.         }
151.         return array;
152.     }
153.     /**
154.      * A procedure that will print an array to the console.
155.      * @param array The array to be printed.
156.      */
157.     public static void printArray(int[] array){
158.         System.out.print("{ ");
159.         for (int i=0; i<array.length; i++){
160.             System.out.print(array[i]);
161.             if (i<array.length-1){ //depicts an array format in the console
162.                 System.out.print(", ");
163.             }
164.         }
165.         System.out.print(" }\n");
166.     }
167.     /**
168.      * A method that takes an array and finds the minimum and maximum values.
169.      * @param array The array for which we want to find the min and max for.
170.      * @return An array that holds the minimum as the first value and maximum as the second value.
171.      */
172.     public static int[] findMinAndMax(int[] array){
173.         int min = array[0];
174.         int max = array[0];
175.         for (int i=0; i<array.length; i++){
176.             if (array[i]>max){
177.                 max = array[i];
178.             }
179.             if (array[i]<min){
180.                 min = array[i];
181.             }
182.         }
183.         int[] minAndMax = new int[]{min, max};
184.         return minAndMax;
185.     }
186.     /**

```

```

187.  * A method that finds the average value from the array.
188.  * @param array The array for which we want to find the average.
189.  * @return The average value.
190.  */
191.  public static double findAverage(int[] array){
192.      double sum = 0;
193.      for (int i=0; i<array.length; i++){
194.          sum += array[i];
195.      }
196.      return sum/(double)array.length;
197.  }
198.  /**
199.   * A method that checks if two sets overlap.
200.   * @param array1 First set.
201.   * @param array2 Second set.
202.   * @return True if they overlap, false otherwise.
203.   */
204.  public static boolean doSetsOverlap(int[] array1, int[] array2){
205.      int[] minAndMax1 = findMinAndMax(array1); //calls upon another function to find the min and max
206.      int[] minAndMax2 = findMinAndMax(array2);
207.      //System.out.println(minAndMax1[0]);
208.      //System.out.println(minAndMax1[1]);
209.      //System.out.println(minAndMax2[0]);
210.      //System.out.println(minAndMax2[1]);
211.      return minAndMax1[1]>minAndMax2[0] && minAndMax1[0] < minAndMax2[1]; //condition if they overlap
212.  }
213.  /**
214.   * A method that will print the intersection of two sets if it is nonempty or will print 'the empty set
   * if empty.
215.   * @param arrayA First set.
216.   * @param arrayB Second set.
217.   * @return intersection set
218.   */
219.  public static int[] intersection(int[] arrayA, int[] arrayB){
220.      int intersectionSize = 0;
221.      int[] emptySet = new int[0];
222.      //first loops to first find the number of matches in order to declare the intersection array
223.      for (int a=0; a<arrayA.length; a++){
224.          for (int b=0; b<arrayB.length; b++){
225.              if (arrayA[a]==arrayB[b]){ //if match
226.                  intersectionSize++; //increase the count
227.              }
228.          }
229.      }
230.      // now that we know how big the array needs to be we can now start writing it
231.      int[] intersection = new int[intersectionSize];
232.      if (intersectionSize==0){
233.          return emptySet;
234.      } else {
235.          //second loops to now find matches and put them into the array declared
236.          int x = 0; //variable that cycles through the array that is being read to
237.          for (int a=0; a<arrayA.length; a++){
238.              for (int b=0; b<arrayB.length; b++){
239.                  if (arrayA[a]==arrayB[b]){
240.                      intersection[x] = arrayA[a];
241.                      x++;
242.                  }
243.              }
244.          }
245.          return intersection;
246.      }
247.  }
248.  /**
249.   * A method that will find the setDifference of two sets if it exists (will print 'the empty set' if em
   * pty).
250.   * @param arrayA first set
251.   * @param arrayB second set
252.   * @return the set difference

```

```

253.  */
254.  public static int[] setDifference(int[] arrayA, int[] arrayB){
255.      int[] intersection = intersection(arrayA, arrayB); //calls previous function
256.      int[] setDifference = new int[arrayA.length-intersection.length]; //set theory
257.      //System.out.println(arrayA.length+" ", "+", "+intersection.length");
258.      int k = 0;
259.      boolean found = true;
260.      /**
261.       * Following loop will cycle through the array, and through the intersection.
262.       * We know that if the element is present in the intersection, it must exist in the
263.       * alternate set. Therefore it won't be present in the set difference.
264.       */
265.      for (int i=0; i<arrayA.length; i++){
266.          for (int j=0; j<intersection.length; j++){
267.              if (arrayA[i]==intersection[j]){
268.                  found = false;
269.              }
270.          }
271.          if (found){
272.              setDifference[k] = arrayA[i];
273.              k++;
274.          }
275.          found = true;
276.      }
277.      return setDifference;
278.  }
279.  /**
280.   * A procedure that depicts a break in the console. Used to separate the application elements in the co
281.   * nsole.
282.   */
283.   public static void layoutBreak(){
284.       System.out.println("-----");
285.   }

```

# Java Programming and Testing Part 1

---

## GRADEMARK REPORT

---

FINAL GRADE

85/100

GENERAL COMMENTS

### Instructor

#### PART ONE

=====

Engagement with Literature Skills (25) **20**

Knowledge and Understanding Skills (10) **8**

Cognitive and Intellectual Skills (10) **7**

Transferable Skills (5) **4**

Practical Application Skills (50) **46**

### Part 1 Software Testing

-----

The discussion of integration layer testing was detailed and quite interesting. Other aspects of the report were a little brief.

### Code Part1 Number Sets

-----

a. Display numbers in sets COMPLETE

b. Determine max and min COMPLETE

c. Calculate average COMPLETE

d. Determine overlap COMPLETE

e. Intersection COMPLETE

f. Set difference COMPLETE

Use of JUnit YES, THOUGH SLIGHTLY LIMITED

Use of Git YES

Use of Java Doc YES

Very good code - appropriate use of comments, git and some (slightly brief) JUnit testing.

## **PART TWO**

=====

Engagement with Literature Skills (25) **18**

Knowledge and Understanding Skills (10) **8**

Cognitive and Intellectual Skills (10) **7**

Transferable Skills (5) **4**

Practical Application Skills (50) **46**

## **Part 2 Design Principles**

-----

A little brief in places, but a good report overall. Take care to ensure that, if you use web references, you select those with high veracity - for example, reference 4 is fine, Oracle are clearly authoritative when it comes to Java. By contrast, the tutorialspoint references are more tough to have confidence in the credibility of.

## **Code Part 2 Connect 4**

-----

Display board COMPLETE

Enter column COMPLETE

Check column position is empty COMPLETE

Drop counter to position COMPLETE

Check win/lose COMPLETE

Blitz feature COMPLETE

Time bomb feature COMPLETE,

Use of JUnit YES, A BIT BRIEF

Use of Git YES

Use of Java Doc YES

### Overall Comments

-----

Very good code - appropriate use of comments, git and some (slightly brief) JUnit testing.

Marks Part One: **85**

Marks Part Two: **83**

---

PAGE 1

---

PAGE 2

---

PAGE 3

---

PAGE 4



### Comment 1

This very detailed and clear.

Good

---

PAGE 5

---

PAGE 6

---

