CS 471 – HW 5 – Software Testing

For all of the definitions, you may find your answers online. Please give your source when answering. Submit this document with your answers in pdf format. For the coding, submit a zip. For any of the code examples, you must use your own code that you have written. State the relevant class, work project, or independent project for which the code was written. Typically the code you give should not be too lengthy – just one method on a class will suffice most cases. Unless otherwise stated, you may use the same code for more than one question. If you don’t have code handy, you may write code expressly for this assignment.

1. Define the term “edge case testing”. Give an example of code you have written for which you would do edge case testing. State the cases and why they are considered edge cases.

**Edge case is an issue that occurs at an extreme operating parameter. (**[**Source**](https://www.educative.io/edpresso/corner-case-vs-edge-case)**)**

**To test a linked list, an edge case would be deleting the first item, last item, or if the item was the only thing in the list. These are edge cases because they are different the normal case of deleting. It would mean that there is something special about this particular input needs to be handled specifically.**

1. Define “equivalence partitioning” as it applies to software test. Give an example of code you have written, examples of equivalence classes of input, why the classes differ, and how they effectively exercise the code

**Equivalence partitioning is when you divide the input domain into different sets of data that can help tests be derived from. (**[**Source**](https://www.geeksforgeeks.org/equivalence-partitioning-method/)**)**

**Binary search, examples of input classes being, one element in the array with the key in the array, the key not in the array. The key being less than the mid point of the array, and the key being greater than the mid point of the array. Theses different classes of input exercise the code by checking if a key is in the array, if it does not, if it is on the boundary of the mid point either greater or less than.**

1. For the code you used in questions #2, give the black box tests derived from the equivalence partitions.

|  |  |  |
| --- | --- | --- |
| Input Array | Input key | Output |
| 10 | 0 | False |
| 10 | 10 | True |
| 3,10,13,20,30 | 10 | True |
| 3,10,13,20,30 | 20 | True |
| 3,10,13,20,29 | 30 | False |

1. Give an example of a method you have written that involves looping structures. Draw the control graph for the structure (see slide deck titled Testing, starting at slide 84) . Give a test case that will exercise each path through the nodes. For each case, give the sequence of nodes that will be encountered.

**1 public static void sumCalc(){**

**2 sum = 0;**

**3 for (int I = 0; I <= 10; I++){**

**4 sum += I;**

**5 }**

**6 return sum;**

**}**

****

1. For a method you have written, write three mutation tests. Explain how the mutation exercises the code in the context of expected results from the mutations.

**Mutation #1: Change to for(int I = 10; I >= 0; I++)**

**This mutation allows to test if the loop went in the opposite direction without changing how int I increases**

**Mutation #2: Change to for(int I = 10; I <= 0; I--)**

**This mutation allows to test if the loop went in the opposite direction with changing I from increasing to decreasing**

**Mutation #3: Change to for(int I = 0; I <= -10; I--)**

**This mutation allows to test if the loop ended in the negative values with changing how int I increases to decreases.**

1. You will now write a Junit Test for a class you have written. I will show you how to do this in NetBeans but if you have a preferred platform, you may use it as long as you give me instructions on how to set it up and run it in my environment. To assure you get the most from this assignment, please either email me the code for the class or show me in person so I can approve your idea. Your test must test edge cases, cases in each equivalence partition, and test at least four paths through a looping structure. Additionally, you must perform one mutation test. Submit a zip of your code.