**Logbook**

**Bradley Pratt - Computer Games Programming U1664020314**

**Algorithms Processes and Data**

**Week 1-2 :**

**package** intArrays;

**import** java.util.Arrays;

**public** **class** CleverRandomListing **extends** RandomListing {

**public** CleverRandomListing (**int** size) {

**super**(size);

}

**protected** **void** randomise() {

**for** (**int** index = 0; index < getArray().length; index++) {

**int** randomArray = getRandomIndex(); // Uses the getRandomIndex method to randomise the array index

**int** newInt = getArray()[randomArray];

getArray()[randomArray] = getArray()[index]; // Changes the grabbed array to randomise its index

getArray()[index] = newInt; // Builds the array using its new index

}

}

**public** **static** **void** main(String[] args) {

RandomListing count = **new** CleverRandomListing(50); // create a new list, as long as the specified length.

System.***out***.println(Arrays.*toString*(count.getArray())); // prints the array to the console

}

}

**public** **class** CleverRandomListingTest **extends** ListingTest {

**private** **static** **long** *testStart*, *testEnd*;

@BeforeClass

**public** **static** **void** setUpBeforeClass() **throws** Exception {

}

@AfterClass

**public** **static** **void** tearDownAfterClass() **throws** Exception {

}

@Rule **public** TestName testName = **new** TestName();

@Before

**public** **void** setUp() **throws** Exception {

*testStart* = System.*nanoTime*();

}

@After

**public** **void** tearDown() **throws** Exception {

*testEnd* = System.*nanoTime*();

System.***out***.println("Test \"" + testName.getMethodName() + "\" took " + (*testEnd*-*testStart*)/1000 + " microseconds");

}

@Test

**public** **void** testOneSize()[[1]](#footnote-1) {

testSize(1,**new** CleverRandomListing(1));

}

**Week 3-4**

**public** **class** GenericMethods {

/\*\*

\* Check if two objects are equal

\* **@param** object1 the first object

\* **@param** object2 the second object

\* **@return** true if the objects are equal (according to the equals() method)

\*/

**public** **static** <T> **boolean** equals(T object1,T object2) {

**if** (object1==**null**) {

**return** object2==**null**;

} **else** {

**return** object1.equals(object2);

}

}

**public** **static** <T> **void** swap(T[] array, **int** index1, **int** index2) {

T objectOne = array[index1];

T objectTwo = array[index2];

array[index1] = objectTwo; //Uses the defined first position and places "objectTwo" there

array[index2] = objectOne; //Uses the defined second position and places "objectOne" there

}

**public** **static** <T> String max(String[] array, **int** index1, **int** index2) {

**int** index = 0;

**int** elementLength = array[0].length();

System.***out***.println();

**for** (**int** i = 0; i < array.length; i++) {

**if** (i >= index1 && i <= index2) {

**if** (array[i].length() > elementLength) {

index = i;

elementLength = array[i].length();

}

}

}

**return** array[index];

}

**public** **static** String main() {

Integer[] arrayTest = {1,2,3,4,5};

*swap*(arrayTest, 1, 2); //Chooses which objects from the defined index location to swap

System.***out***.println(Arrays.*toString*(arrayTest));

String[] names = {"Hugh", "Andrew", "Ebrahim","Diane","Paula", "Simon"};

System.***out***.println(*max*(names,1,4));

**return** Arrays.*toString*(arrayTest);

}

}

@Test

**public** **void** testSwap() {

*assertEquals*("[1, 3, 2, 4, 5]",GenericMethods.*main*());

}

1. The tests go up to a million, the same as the default test classes. [↑](#footnote-ref-1)