ID	18
Verfahren	Parametrisierter JUnit-Test
Klasse	Landscape
Methoden	getDistance(City city, City neighbour): double Rückgabe der Distanz zweier Städte zueinander
Vorbedingung	City(1), City(2) City(2) hat zu City(1) eine Distanz von 5
Eingaben	City(1), City(2)
Erwartetes Ergebnis	5
Ergebnis	5 **Test erfolgreich**

ID	19
Verfahren	Parametrisierter JUnit-Test
Klasse	Landscape
Methoden	getNeighboursSize(): int Rückgabe der Größe der Distanzmatrix
Vorbedingung	City(1), City(2) City(2) hat zu City(1) eine Distanz von 5
Eingaben	Distanzmatrix hat eine Größe von 2
Erwartetes Ergebnis	Distanzmatrix hat eine Größe von 2
Ergebnis	**Test erfolgreich**

ID	20
Verfahren	Parametrisierter JUnit-Test
Klasse	Landscape
Methoden	getSpecifiedNeighbours(City currentCity): double[] Rückgabe der Distanzen zu allen erreichbaren Städten ausgehend von der übergebenen Stadt
Vorbedingung	City(1), City(2), City(3) Städte haben untereinander eine Distanz von je 1
Eingaben	City(1)
Erwartetes Ergebnis	{1,1}
Ergebnis	{1,1} **Test erfolgreich**

ID	21
Verfahren	Parametrisierter JUnit-Test
Klasse	Landscape
Methoden	getNeighbours(): double[][] Rückgabe der kompletten Distanzmatrix
Vorbedingung	City(1), City(2), City(3) Städte haben untereinander eine Distanz von je 1
Eingaben	-
Erwartetes Ergebnis	Distanzmatrix hat eine Länge von 3
Ergebnis	Distanzmatrix hat eine Länge von 3 **Test erfolgreich**

ID	22
Verfahren	Parametrisierter JUnit-Test
Klasse	Landscape
Methoden	addNeighbours(City a, City b, double distance): int Abspeichern der übergebenen Distanz zwischen a und b in die Pheromonmatrix
Vorbedingung	-
Eingaben	City(1), City(2), 1
Erwartetes Ergebnis	Checksumme ist 1 Distanzmatrix hat eine Länge von 2 Distanz zwischen City(1) und City(2) beträgt 1
Ergebnis	Checksumme ist 1 Distanzmatrix hat eine Länge von 2 Distanz zwischen City(1) und City(2) beträgt 1 **Test erfolgreich**

ID	23
Verfahren	Parametrisierter JUnit-Test
Klasse	Landscape
Methoden	addNeighbours(City a, City b, double distance): int Abspeichern der übergebenen Distanz zwischen a und b in die Pheromonmatrix
Vorbedingung	City(1), City(2) City(1) hat eine Distanz von 1 zu City(2)
Eingaben	City(1), City(2), 1
Erwartetes Ergebnis	Checksumme ist -1 Distanzmatrix hat eine Länge von 2
Ergebnis	Checksumme ist -1 Distanzmatrix hat eine Länge von 2 **Test erfolgreich**