**shortestPath() algorithm function runtime comparison:**

**Ex2 Java project VS. Ex3 Python project**

**checking shortestPath from node with key 0 to a node with key 4**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Ex3- time in sec | Ex2-time in sec | Graph name |
|  | 0.000 | 0.035 | G\_10\_80\_1.json |
|  | 0.001 | 0.035 | G\_100\_800\_1.json | |
|  | 0.230 | 0.158 | G\_1000\_8000\_1.json | |
|  | 34.725 | 1.246 | G\_10000\_80000\_1.json |
|  | 129.806 | 3.849 | G\_20000\_160000\_1.json |
|  | 371.778 | 10.094 | G\_30000\_240000\_1.json |

|  |  |
| --- | --- |
| Networkx- time in sec | Graph Name |
| 0.000 | G\_10\_80\_1.json |
| 0.000 | G\_100\_800\_1.json |
| 0.002 | G\_1000\_8000\_1.json |
| 0.228 | G\_10000\_80000\_1.json |
| 0.521 | G\_20000\_160000\_1.json |
| 0.613 | G\_30000\_240000\_1.json |

נכונות האלגוריתמים: בכל האלגוריתמים יצאו אותן התוצאות:

|  |  |  |
| --- | --- | --- |
| Weight of path | List | Graph name |
| 1.4254324246464245 | [0, 4] | G\_10\_80\_1.json |
| 29.14407437224567 | [0, 1, 4] | G\_100\_800\_1.json |
| 27.00569088191026 | [0, 371, 233, 4] | G\_1000\_8000\_1.json |
| 99.2798668896577 | [0, 9624, 2121, 8592, 3136, 7829, 6805, 4] | G\_10000\_80000\_1.json |
| 125.17665968464343 | [0, 5855, 4630, 352, 10238, 11183, 4872, 6874, 9707, 7676, 4] | G\_20000\_160000\_1.json |
| 123.37112242332906 | [0, 27764, 5574, 25737, 16604, 18307, 15273, 19586, 12002, 14489, 14370, 12768, 3852, 4] | G\_30000\_240000\_1.json |

@Test  
void test\_javaComparisonJson1() throws IOException {  
 System.*out*.println("reading from file..."+ag1.load("G\_10\_80\_1.json"));  
 System.*out*.println("shortestPath is "+ag1.shortestPathDist(0, 4));  
}  
  
@Test  
void test\_javaComparisonJson2() throws IOException {  
 System.*out*.println("reading from file..."+ag1.load("G\_100\_800\_1"));  
 System.*out*.println("shortestPath is "+ag1.shortestPathDist(0, 4));  
}  
  
@Test  
void test\_javaComparisonJson3() throws IOException {  
 System.*out*.println("reading from file..."+ag1.load("G\_1000\_8000\_1"));  
 System.*out*.println("shortestPath is "+ag1.shortestPathDist(0, 4));  
 }  
  
@Test  
void test\_javaComparisonJson4() throws IOException {  
 System.*out*.println("reading from file..."+ag1.load("G\_10000\_80000\_1"));  
 System.*out*.println("shortestPath is "+ag1.shortestPathDist(0, 4));  
 }  
  
@Test  
void test\_javaComparisonJson5() throws IOException {  
 System.*out*.println("reading from file..."+ag1.load("G\_20000\_160000\_1"));  
 System.*out*.println("shortestPath is "+ag1.shortestPathDist(0, 4));  
 }  
  
@Test  
void test\_javaComparisonJson6() throws IOException {  
 System.*out*.println("reading from file..."+ag1.load("G\_30000\_240000\_1"));  
 System.*out*.println("shortestPath is "+ag1.shortestPathDist(0, 4));  
}

**Ex2:**

**Setting up the environment. creating the graphs...**

**operating on file: G\_10\_80\_1.json**

**reading from file...true**

**shortestPathList is [0, 4]**

**shortestPathDist is 1.4254324246464245**

**This test took 0.049 seconds**

**Setting up the environment. creating the graphs...**

**operating on file: G\_100\_800\_1.json**

**reading from file...true**

**shortestPathList is [0, 1, 4]**

**shortestPathDist is 29.14407437224567**

**This test took 0.052 seconds**

**Setting up the environment. creating the graphs...**

**operating on file: G\_1000\_8000\_1.json**

**reading from file...true**

**shortestPathList is [0, 371, 233, 4]**

**shortestPathDist is 27.00569088191026**

**This test took 0.224 seconds**

**Setting up the environment. creating the graphs...**

**operating on file: G\_10000\_80000\_1.json**

**reading from file...true**

**shortestPathList is [0, 9624, 2121, 8592, 3136, 7829, 6805, 4]**

**shortestPathDist is 99.2798668896577**

**This test took 1.823 seconds**

**Setting up the environment. creating the graphs...**

**operating on file: G\_20000\_160000\_1.json**

**reading from file...true**

**shortestPathList is [0, 5855, 4630, 352, 10238, 11183, 4872, 6874, 9707, 7676, 4]**

**shortestPathDist is 125.17665968464343**

**This test took 4.823 seconds**

**Setting up the environment. creating the graphs...**

**operating on file: G\_30000\_240000\_1.json**

**reading from file...true**

**shortestPathList is [0, 27764, 5574, 25737, 16604, 18307, 15273, 19586, 12002, 14489, 14370, 12768, 3852, 4]**

**shortestPathDist is 123.37112242332906**

**This test took 8.718 seconds**