File 1)

```
jehad@jehad-MacBookPro:~/Downloads/Ass3$ make
g++ Dijkstra.cpp Edge.cpp Graph.cpp List_Graph.cpp Matrix_Graph.cpp -o a.out -std=c++11
jehad@jehad-MacBookPro:~/Downloads/Ass3$ ./a.out test1.txt Matrix
This is the Standard Dijkstra's Algorithm...
The Standard Dijkstra's Algorithm for Matrix took 10 microseconds.
Results for the Standard Dijkstra's Algorithm:
Vertex Distance from Source Predecessor
      0
         10
                              0
3
                              0
Running priority queue-based Dijkstra's algorithm...
Priority Queue-based Dijkstra's algorithm for Matrix took 17 microseconds.
Results for the Priority Queue-based Dijkstra's Algorithm:
Vertex Distance from Source Predecessor
       0
         10
                              0
         50
3
         30
                              0
         60
Comparing results of both implementations:
Standard Dijkstra's algorithm is faster.
Both implementations produced the same results.
jehad@jehad-MacBookPro:~/Downloads/Ass3$ ./a.out test1.txt List
This is the Standard Dijkstra's Algorithm...
The Standard Dijkstra's Algorithm for List took 13 microseconds.
Results for the Standard Dijkstra's Algorithm:
Vertex Distance from Source Predecessor
0
1
                                 0
2
         50
                                 3
3
          30
                                 0
4
                                 2
          60
Running priority queue-based Dijkstra's algorithm...
Priority Queue-based Dijkstra's algorithm for List took 21 microseconds.
Results for the Priority Queue-based Dijkstra's Algorithm:
Vertex Distance from Source Predecessor
        0
10
                               -1
                                0
2
         50
3
          30
                                 0
4
          60
                                 2
Comparing results of both implementations:
Standard Dijkstra's algorithm is faster.
Both implementations produced the same results.
```

File 2)

```
This is the Standard Dijkstra's Algorithm...
The Standard Dijkstra's Algorithm for Matrix took 20 microseconds.
Results for the Standard Dijkstra's Algorithm:
Vertex Distance from Source Predecessor
   0 -1
8 3
9 1
                               0
                               0
Running priority queue-based Dijkstra's algorithm...
Priority Queue-based Dijkstra's algorithm for Matrix took 16 microseconds.
Results for the Priority Queue-based Dijkstra's Algorithm:
Vertex Distance from Source Predecessor
    0 -1
8 3
2
                               0
Comparing results of both implementations:
Priority Queue-based Dijkstra's algorithm is faster.
Both implementations produced the same results
This is the Standard Dijkstra's Algorithm...
The Standard Dijkstra's Algorithm for List took 16 microseconds.
Results for the Standard Dijkstra's Algorithm:
Vertex Distance from Source Predecessor
      0
                            0
Running priority queue-based Dijkstra's algorithm...
Priority Queue-based Dijkstra's algorithm for List took 17 microseconds.
Results for the Priority Queue-based Dijkstra's Algorithm:
Vertex Distance from Source Predecessor
3
                            0
                             0
Comparing results of both implementations:
Standard Dijkstra's algorithm is faster.
Both implementations produced the same results.
```

```
This is the Standard Dijkstra's Algorithm...
The Standard Dijkstra's Algorithm for Matrix took 20 microseconds.
Results for the Standard Dijkstra's Algorithm:
Vertex Distance from Source Predecessor
          320
          450
           570
           630
           610
           680
           800
Running priority queue-based Dijkstra's algorithm...
Priority Queue-based Dijkstra's algorithm for Matrix took 47 microseconds.
Results for the Priority Queue-based Dijkstra's Algorithm:
          Distance from Source Predecessor
Vertex
          320
           500
           630
           610
           680
           800
Comparing results of both implementations:
Standard Dijkstra's algorithm is faster.
Both implementations produced the same results.
This is the Standard Dijkstra's Algorithm...
The Standard Dijkstra's Algorithm for List took 30 microseconds.
Results for the Standard Dijkstra's Algorithm:
Vertex Distance from Source Predecessor
          0
           320
           500
           570
           630
           610
           680
           800
Running priority queue-based Dijkstra's algorithm...
Priority Queue-based Dijkstra's algorithm for List took 17 microseconds.
Results for the Priority Queue-based Dijkstra's Algorithm:
Vertex Distance from Source Predecessor
           320
           450
2
4
5
6
           500
           570
           630
           610
           680
           800
           860
Comparing results of both implementations:
Priority Queue-based Dijkstra's algorithm is faster.
```

Both implementations produced the same results.

File 4)

```
Running priority queue-based Dijkstra's algorithm...
Priority Queue-based Dijkstra's algorithm for Matrix took 37 microseconds.
Results for the Priority Queue-based Dijkstra's Algorithm:
                    Distance from Source Predecessor
                                  ice from Source
-1
 Vertex
                    0
320
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
20
21
22
23
24
                                                                  0
1
1
2
4
4
3
7
7
9
10
10
12
13
14
15
16
18
19
20
21
23
                    450
500
570
630
610
680
800
                     860
1000
                     1100
1130
                     1250
1360
1465
                     1510
1580
                      1600
                     1660
1740
                     1885
1880
                      1980
Comparing results of both implementations:
Priority Queue-based Dijkstra's algorithm is faster.
Both implementations produced the same results.
```

jehad@jehad-MacBookPro:~/Downloads/Ass3\$./a.out test4.txt List This is the Standard Dijkstra's Algorithm... The Standard Dijkstra's Algorithm for List took 172 microseconds. Results for the Standard Dijkstra's Algorithm: Vertex Distance from Source Predecessor 320 -1 0 1 1 2 4 4 3 7 7 9 10 10 12 13 14 15 16 18 19 20 21 23 630 10 11 12 13 14 15 16 17 18 20 21 22 23 24 1130 1360 1980

Running priority queue-based Dijkstra's algorithm... Priority Queue-based Dijkstra's algorithm for List took 45 microseconds. Results for the Priority Queue-based Dijkstra's Algorithm:

		,	J	9
Vartov	Distance from	Course	Dradacaccar	

Vertex	Distance from Source	Predecessor
0	0	-1
1	320	0
1 2	450	1
3	500	1
4	570	2
5	630	4
6	610	4
7	680	3
8	800	7
9	860	7
10	1000	9
11	1100	10
12	1130	10
13	1250	12
14	1360	13
15	1465	14
16	1510	14
17	1580	15
18	1600	16
19	1660	18
20	1740	18
21	1785	19
22	1885	20
23	1880	21
24	1980	23

Comparing results of both implementations:

Priority Queue-based Dijkstra's algorithm is faster.

Both implementations produced the same results.