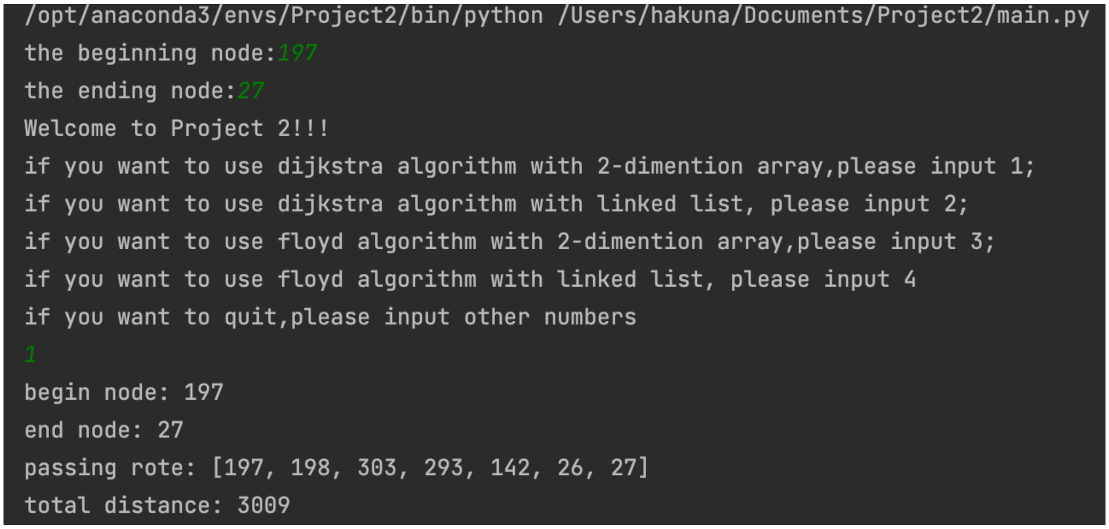
Name: Yijun Zhou
Student ID: 4509000
Email: yiz209@pitt.edu

Project2-Algorithms

1.output of test case

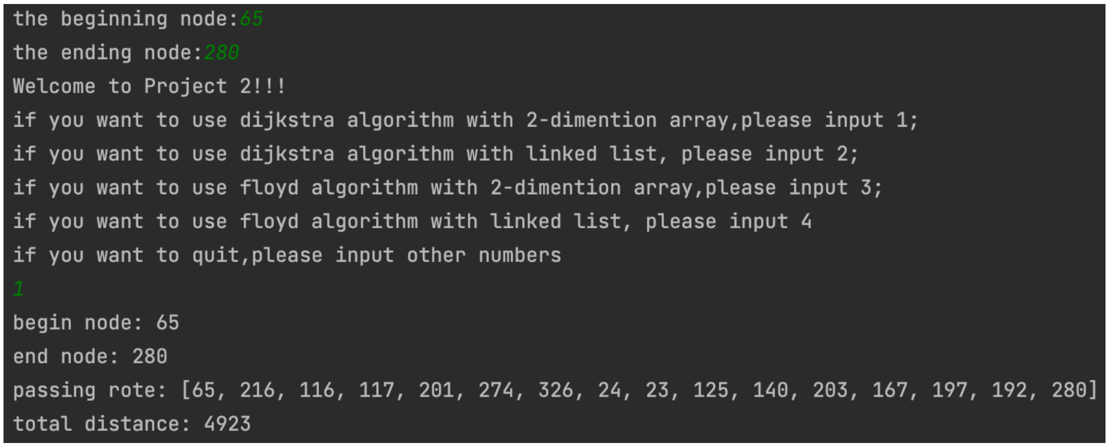
- Algorithm 1-dijkstra algorithm(2-array)

  Test case 1:



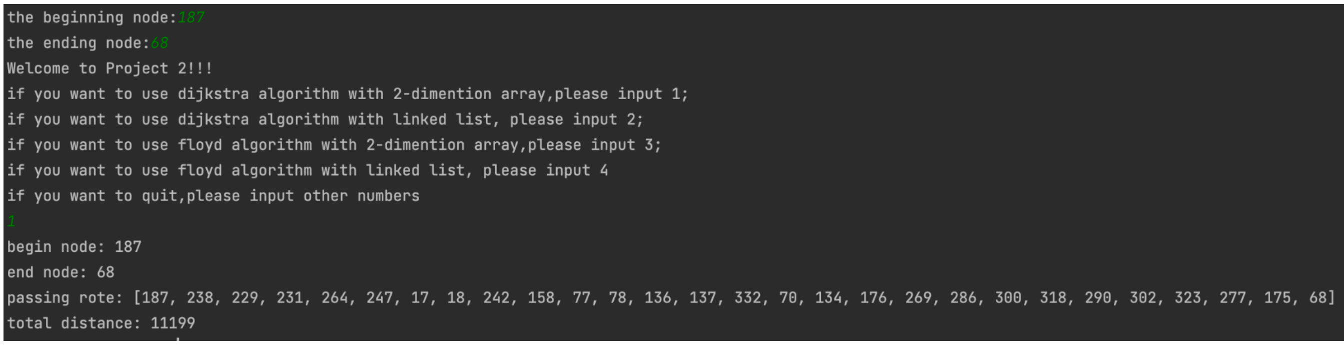
```
/opt/anaconda3/envs/Project2/bin/python /Users/hakuna/Documents/Project2/main.py
the beginning node:197
the ending node:27
Welcome to Project 2!!!
if you want to use dijkstra algorithm with 2-dimention array,please input 1;
if you want to use dijkstra algorithm with linked list, please input 2;
if you want to use floyd algorithm with 2-dimention array,please input 3;
if you want to use floyd algorithm with linked list, please input 4
if you want to quit,please input other numbers
1
begin node: 197
end node: 27
passing rote: [197, 198, 303, 293, 142, 26, 27]
total distance: 3009
```

  Test case 2:



```
the beginning node:65
the ending node:280
Welcome to Project 2!!!
if you want to use dijkstra algorithm with 2-dimention array,please input 1;
if you want to use dijkstra algorithm with linked list, please input 2;
if you want to use floyd algorithm with 2-dimention array,please input 3;
if you want to use floyd algorithm with linked list, please input 4
if you want to quit,please input other numbers
1
begin node: 65
end node: 280
passing rote: [65, 216, 116, 117, 201, 274, 326, 24, 23, 125, 140, 203, 167, 197, 192, 280]
total distance: 4923
```

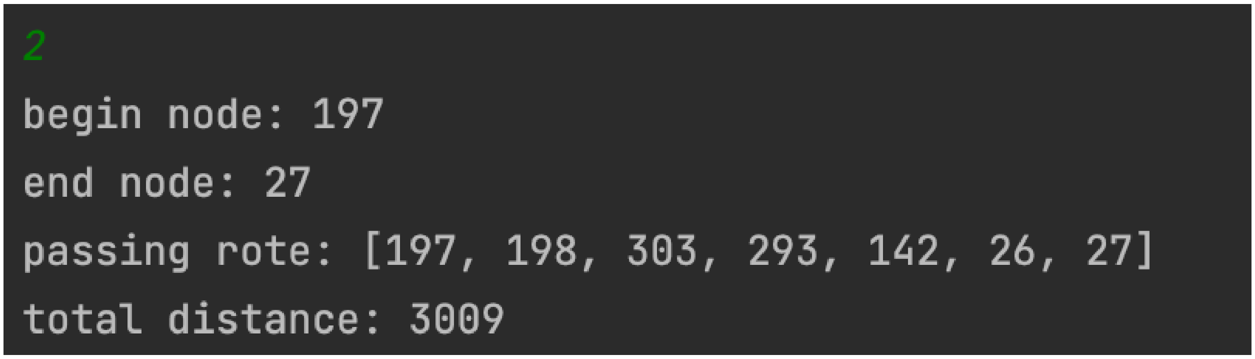
Test case 3:



```
the beginning node:187
the ending node:68
Welcome to Project 2!!!
if you want to use dijkstra algorithm with 2-dimention array,please input 1;
if you want to use dijkstra algorithm with linked list, please input 2;
if you want to use floyd algorithm with 2-dimention array,please input 3;
if you want to use floyd algorithm with linked list, please input 4
if you want to quit,please input other numbers
1
begin node: 187
end node: 68
passing rote: [187, 238, 229, 231, 264, 247, 17, 18, 242, 158, 77, 78, 136, 137, 332, 70, 134, 176, 269, 286, 300, 318, 290, 302, 323, 277, 175, 68]
total distance: 11199
```

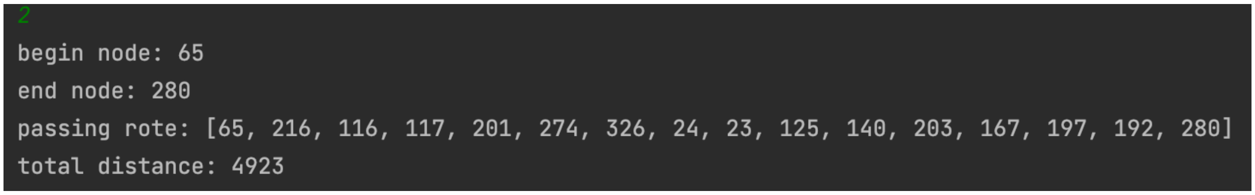
- Algorithm 2-dijkstra algorithm(linkedlist)

Test case 1:



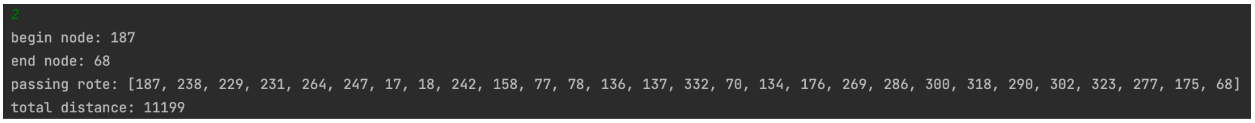
```
2
begin node: 197
end node: 27
passing rote: [197, 198, 303, 293, 142, 26, 27]
total distance: 3009
```

Test case 2:



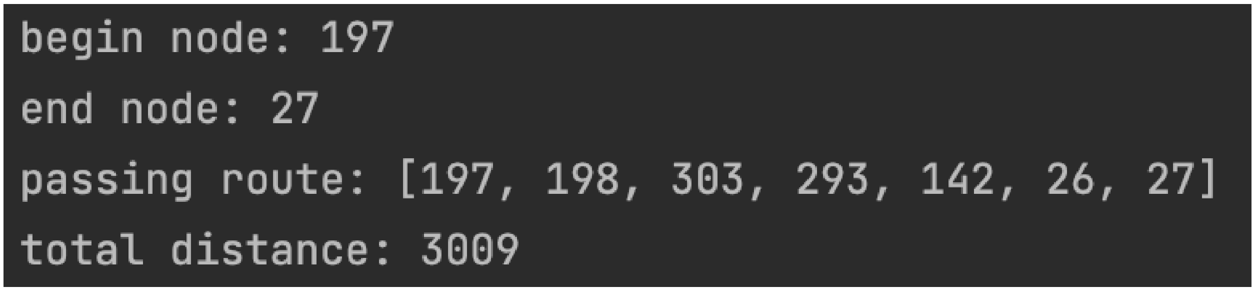
```
2
begin node: 65
end node: 280
passing rote: [65, 216, 116, 117, 201, 274, 326, 24, 23, 125, 140, 203, 167, 197, 192, 280]
total distance: 4923
```

Test case 3:



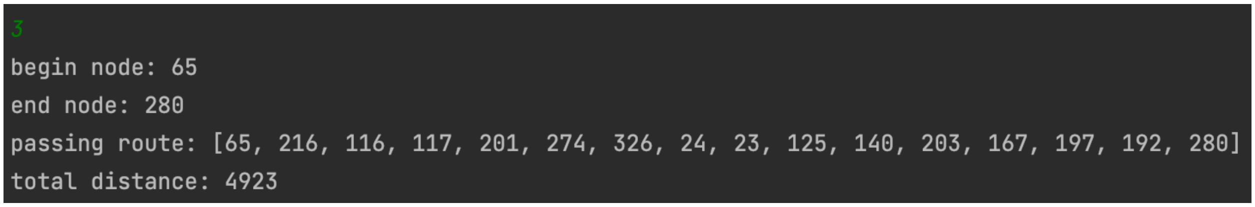
```
2
begin node: 187
end node: 68
passing rote: [187, 238, 229, 231, 264, 247, 17, 18, 242, 158, 77, 78, 136, 137, 332, 70, 134, 176, 269, 286, 300, 318, 290, 302, 323, 277, 175, 68]
total distance: 11199
```

- Algorithm 3-Floyd’s algorithm(2-array)
  Test case 1:



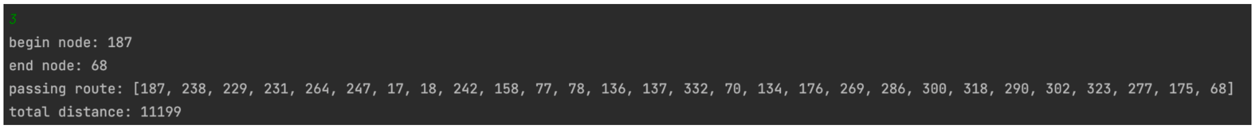
```
begin node: 197
end node: 27
passing route: [197, 198, 303, 293, 142, 26, 27]
total distance: 3009
```

Test case 2:



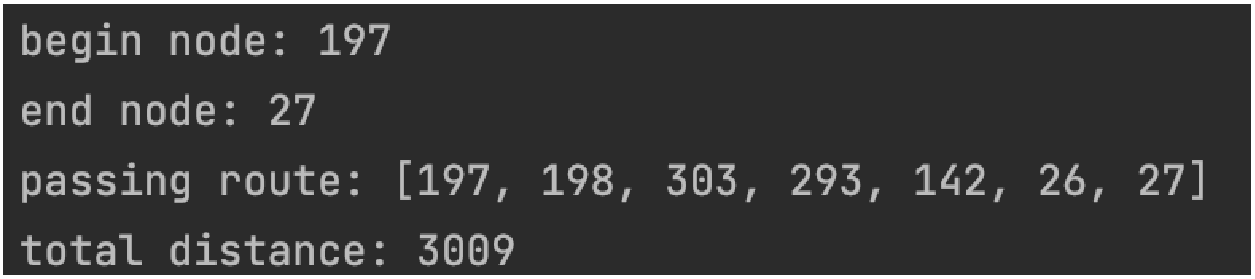
```
3
begin node: 65
end node: 280
passing route: [65, 216, 116, 117, 201, 274, 326, 24, 23, 125, 140, 203, 167, 197, 192, 280]
total distance: 4923
```

Test case 3:



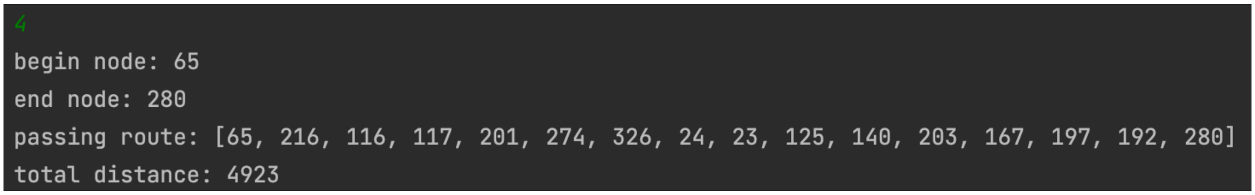
```
3
begin node: 187
end node: 68
passing route: [187, 238, 229, 231, 264, 247, 17, 18, 242, 158, 77, 78, 136, 137, 332, 70, 134, 176, 269, 286, 300, 318, 290, 302, 323, 277, 175, 68]
total distance: 11199
```

- Algorithm 4-Floyd’s algorithm(linkedlist)
  Test case 1:



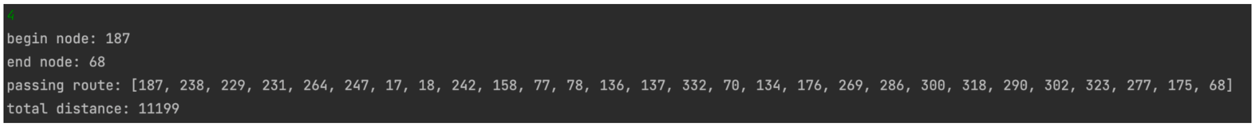
```
begin node: 197
end node: 27
passing route: [197, 198, 303, 293, 142, 26, 27]
total distance: 3009
```

Test case 2:



```
4
begin node: 65
end node: 280
passing route: [65, 216, 116, 117, 201, 274, 326, 24, 23, 125, 140, 203, 167, 197, 192, 280]
total distance: 4923
```

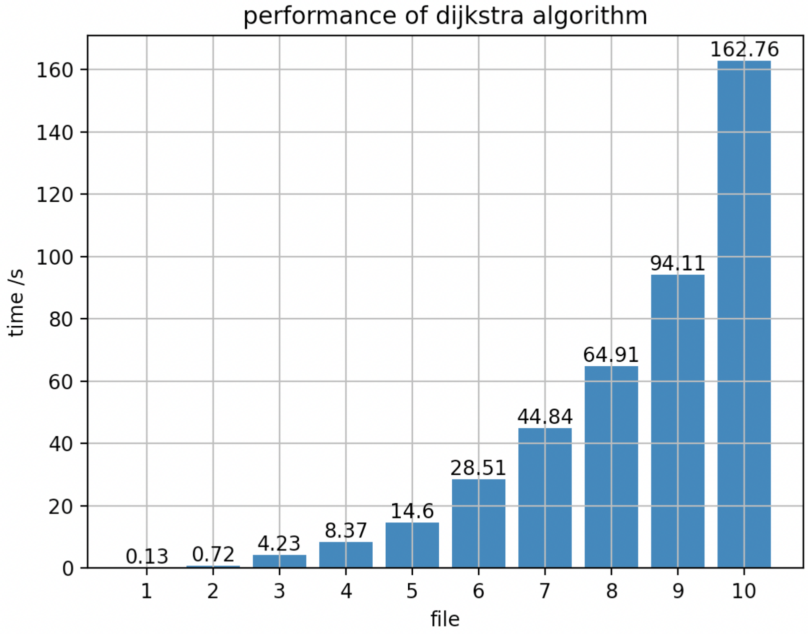
Test case 3:



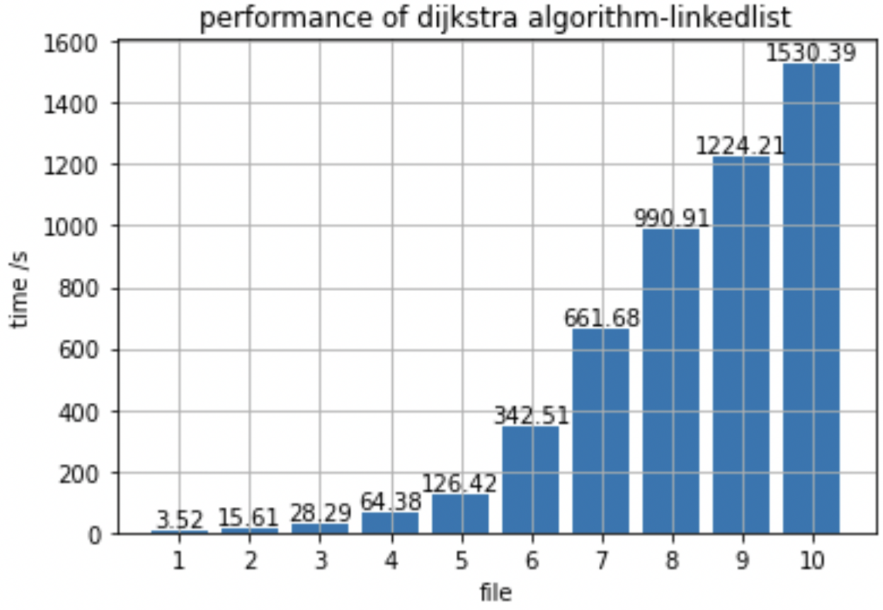
```
4
begin node: 187
end node: 68
passing route: [187, 238, 229, 231, 264, 247, 17, 18, 242, 158, 77, 78, 136, 137, 332, 70, 134, 176, 269, 286, 300, 318, 290, 302, 323, 277, 175, 68]
total distance: 11199
```

2.output of performance

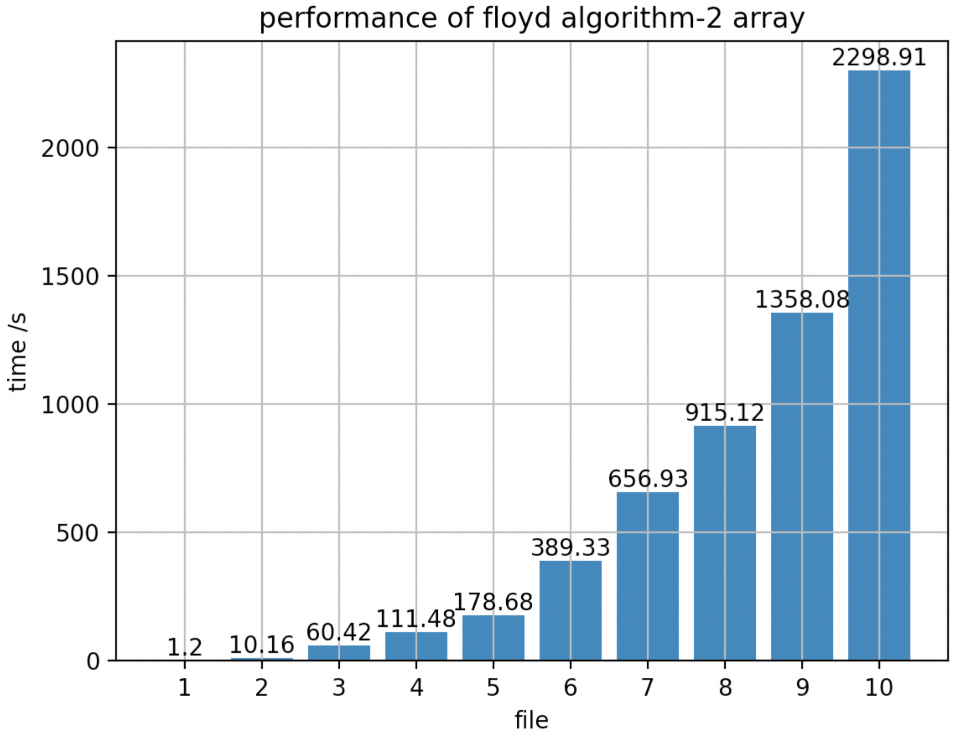
- Algorithm 1-dijkstra algorithm(2-array)



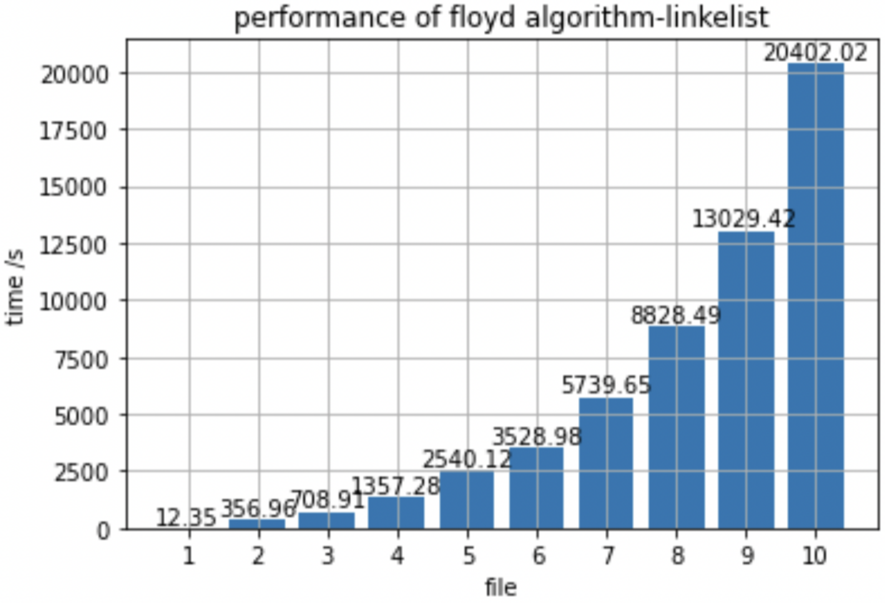
- Algorithm 2-dijkstra algorithm(linkedlist)



- Algorithm 3-Floyd’s algorithm(2-array)



- Algorithm 4-Floyd’s algorithm(linkedlist)


performance of floyd algorithm-linkelist
time /s
file