Data

```
In[ ]:= vertexcoords = {{36.97984, -86.448705}, {36.97333, -86.477607},
    {36.979892, -86.467661}, {36.98153, -86.47028}, {36.977102, -86.462463},
    {36.978808, -86.465039}, {36.9866438, -86.455633}, {36.978752, -86.479432},
    {36.981265, -86.477447}, {36.978159, -86.458263}, {36.981901, -86.461781},
    {36.978313, -86.484294}, {36.978523, -86.482785}, {36.979114, -86.481927},
    {36.979997, -86.481075}, {36.975545, -86.448493}, {36.9888659, -86.457797},
    {36.9653648, -86.4877243}, {36.984872, -86.4577918}, {36.981278, -86.486818},
    {36.965689, -86.484376}, {36.967673, -86.486414}, {36.968679, -86.487533},
    {36.964376, -86.453163}, {36.964023, -86.455322}, {36.980242, -86.471767},
    {36.980346, -86.471752}, {36.990373, -86.463696}, {36.990641, -86.459695},
    {36.985013, -86.462676}, {36.986466, -86.462399}, {36.97936, -86.448788},
    {36.975466, -86.474332}, {36.976548, -86.473031}, {36.988282, -86.478283},
    {36.988028, -86.478176}, {36.985751, -86.482864}, {36.9846925, -86.4827997},
    {36.9848469, -86.4821452}, {36.9844502, -86.4806308},
    {36.990062, -86.44996}, {36.9907559, -86.4489183}, {36.972273, -86.466697},
    {36.971154, -86.467519}, {36.969583, -86.46957}, {36.969474, -86.469679},
    {36.97894, -86.453638}, {36.968697, -86.47047}, {36.979659, -86.453348},
    {36.963054, -86.472825}, {36.963352, -86.473322}, {36.9638329, -86.4867821},
    {36.96444, -86.47508}, {36.961592, -86.463258}, {36.963404, -86.463695},
    {36.963191, -86.464782}, {36.963614, -86.466471}, {36.963735, -86.466765},
    {36.975889, -86.480113}, {36.964718, -86.464136}, {36.988475, -86.470635},
    {36.9883855, -86.4702814}, {36.98931, -86.477481}, {36.9891889, -86.4594376},
    {36.9857291, -86.4512571}, {36.9876083, -86.4484356}, {36.96834, -86.470821},
    {36.987116, -86.477687}, {36.975484, -86.474851}, {36.975201, -86.475216},
    {36.97309, -86.477905}, {36.986907, -86.477441}, {36.970037, -86.464673},
    {36.970333, -86.468658}, {36.9665999, -86.4877667}, {36.971676, -86.476781},
    {36.972257, -86.477543}, {36.977867, -86.449021}, {36.967718, -86.481764},
    {36.966672, -86.480485}, {36.988963, -86.456785}, {36.986592, -86.461259},
    {36.981774, -86.457076}, {36.990179, -86.454851}, {36.977008, -86.448253},
    {36.977106, -86.449164}, {36.968555, -86.480697}, {36.978213, -86.484756},
    {36.9837162, -86.4801249}, {36.979287, -86.484958}, {36.979442, -86.484997},
    {36.980195, -86.485169}, {36.9822851, -86.4854919}, {36.959679, -86.454523},
    {36.960452, -86.453145}, {36.965618, -86.466854}, {36.9854178, -86.4863054},
    {36.983226, -86.467445}, {36.982894, -86.46674}, {36.982654, -86.46652},
    {36.983071, -86.466431}, {36.981934, -86.47647}, {36.986895, -86.487461},
    {36.9601778, -86.4480824}, {36.987925, -86.449275}, {36.968352, -86.457701},
    {36.978232, -86.450494}, {36.978342, -86.448947}, {36.980534, -86.482206},
    {36.971983, -86.472735}, {36.975093, -86.469121}, {36.963081, -86.4556419},
    {36.983776, -86.466844}, {36.985645, -86.466132}, {36.986621, -86.46589},
    {36.977345, -86.455553}, {36.9849586, -86.4481738}, {36.9787226, -86.4557158},
    {36.9781737, -86.4557676}, {36.9783662, -86.4554268}, {36.9782062, -86.4556997},
    {36.982524, -86.487499}, {36.9774883, -86.4567729}, {36.9774059, -86.4567385},
    {36.9772694, -86.4563696}, {36.9777744, -86.4565804}, {36.9773017, -86.4562686},
    {36.9773942, -86.4559518}, {36.9778163, -86.4571561}, {36.9777053, -86.4560508},
    {36.971996, -86.486029}, {36.971722, -86.487659}, {36.9606, -86.454463},
    {36.9618139, -86.4668643}, {36.97034, -86.48684}, {36.9680696, -86.4851531},
    {36.9705, -86.471889}, {36.965441, -86.474341}, {36.969218, -86.471607},
    {36.980361, -86.484085}, {36.959021, -86.462467}, {36.964902, -86.453964},
```

```
{36.975907, -86.487719}, {36.975826, -86.473898}, {36.9748, -86.468734},
{36.976236, -86.474298}, {36.974877, -86.481057}, {36.967952, -86.468528},
{36.975181, -86.480772}, {36.976595, -86.479471}, {36.977311, -86.478833},
{36.978248, -86.478003}, {36.983587, -86.467072}, {36.987787, -86.450742},
{36.989726, -86.447825}, {36.986803, -86.467037}, {36.98771, -86.466849},
{36.973, -86.475109}, {36.97397, -86.476798}, {36.977141, -86.482265},
{36.970374, -86.484312}, {36.969736, -86.479212}, {36.968699, -86.477704},
{36.9775178, -86.4561076}, {36.9777788, -86.456317}, {36.9775868, -86.4561154},
{36.977846, -86.481632}, {36.97643, -86.482912}, {36.9812925, -86.4614799},
{36.988446, -86.449754}, {36.971721, -86.487154}, {36.972837, -86.488364},
{36.981004, -86.451008}, {36.981142, -86.452157}, {36.984307, -86.485399},
{36.981727, -86.453526}, {36.980284, -86.448637}, {36.980409, -86.44988},
{36.980535, -86.45109}, {36.981787, -86.449657}, {36.988624, -86.471842},
{36.978457, -86.4654684}, {36.96653, -86.456308}, {36.972798, -86.471833},
{36.973585, -86.472907}, {36.970412, -86.455426}, {36.967825, -86.481631},
{36.968813, -86.482694}, {36.967914, -86.481519}, {36.977036, -86.4521409},
{36.970063, -86.485137}, {36.970595, -86.485584}, {36.971853, -86.486467},
{36.973271, -86.48699}, {36.983752, -86.484373}, {36.9836895, -86.4795419},
{36.9842021, -86.4790809}, {36.959378, -86.449301}, {36.958937, -86.449775},
{36.959012, -86.449861}, {36.975024, -86.46629}, {36.985464, -86.4878421},
{36.9841323, -86.488258}, {36.9838409, -86.4875285}, {36.976996, -86.487798},
{36.970813, -86.450272}, {36.974737, -86.449564}, {36.97488, -86.451082},
{36.9904521, -86.4615107}, {36.958673, -86.470691}, {36.9903295, -86.4533631},
{36.9852263, -86.4522874}, {36.9813202, -86.4554996}, {36.979707, -86.463959},
{36.9898795, -86.4528565}, {36.988957, -86.456187}, {36.988213, -86.457005},
{36.982495, -86.480348}, {36.981838, -86.481371}, {36.961559, -86.462957},
{36.990778, -86.457035}, {36.990516, -86.456588}, {36.987955, -86.479613},
{36.988415, -86.474452}, {36.989532, -86.472022}, {36.989644, -86.471348},
{36.989645, -86.471257}, {36.989553, -86.47047}, {36.989464, -86.469856},
{36.989375, -86.469275}, {36.989332, -86.468985}, {36.989163, -86.467943},
{36.989001, -86.466976}, {36.98894, -86.466604}, {36.988737, -86.465347},
{36.988427, -86.463384}, {36.988358, -86.462041}, {36.988445, -86.460881},
{36.988612, -86.459717}, {36.988662, -86.459339}, {36.972921, -86.448464},
{36.97312, -86.44984}, {36.973486, -86.452648}, {36.980956, -86.454159},
{36.9843929, -86.4517275}, {36.981633, -86.448404}, {36.981936, -86.450864},
{36.9851163, -86.4506594}, {36.98555, -86.448557}, {36.989397, -86.4509561},
{36.9902888, -86.4521874}, {36.981519, -86.465894}, {36.9887798, -86.4583742},
{36.9876642, -86.4585498}, {36.981763, -86.466292}, {36.976615, -86.44925},
{36.976834, -86.450694}, {36.972095, -86.454306}, {36.972492, -86.454244},
{36.973659, -86.453968}, {36.974861, -86.45376}, {36.975282, -86.453692},
{36.97609, -86.453564}, {36.977133, -86.45339}, {36.977957, -86.453273},
{36.973702, -86.448797}, {36.97464, -86.448644}, {36.974769, -86.448618},
{36.9760554, -86.4832592}, {36.962335, -86.482526}, {36.980905, -86.475085},
{36.980456, -86.475561}, {36.968616, -86.458445}, {36.98151, -86.47579},
{36.980972, -86.476431}, {36.9602057, -86.451599}, {36.9619626, -86.4510672},
{36.971854, -86.458649}, {36.986088, -86.478513}, {36.978856, -86.479037},
{36.976457, -86.475664}, {36.976813, -86.475246}, {36.977358, -86.474618},
{36.970884, -86.47521}, {36.972934, -86.470131}, {36.971736, -86.474531},
{36.972686, -86.473734}, {36.967022, -86.470896}, {36.976045, -86.470289},
{36.97714, -86.468422}, {36.977602, -86.466812}, {36.976544, -86.474765},
{36.976114, -86.475295}, {36.977038, -86.47415}, {36.976825, -86.473567},
```

```
{36.975806, -86.47481}, {36.980204, -86.481645}, {36.978731, -86.48484},
{36.974178, -86.456905}, {36.968736, -86.453578}, {36.969009, -86.455865},
{36.990772, -86.458126}, {36.9898287, -86.4579684}, {36.989384, -86.457882},
{36.969526, -86.457447}, {36.970323, -86.459728}, {36.970693, -86.460764},
{36.971159, -86.461994}, {36.958742, -86.46021}, {36.982739, -86.46795},
{36.983309, -86.46892}, {36.979384, -86.485507}, {36.961139, -86.45916},
{36.984545, -86.470496}, {36.985604, -86.471515}, {36.985895, -86.471792},
{36.986507, -86.472373}, {36.989131, -86.480005}, {36.988774, -86.486202},
{36.9845978, -86.4529322}, {36.9623, -86.453441}, {36.978879, -86.448867},
{36.962866, -86.454614}, {36.989659, -86.459529}, {36.9706724, -86.4843562},
{36.973937, -86.456279}, {36.966295, -86.458257}, {36.967375, -86.453931},
{36.966026, -86.450157}, {36.965008, -86.449127}, {36.964259, -86.448283},
{36.983, -86.448186}, {36.9881371, -86.4528405}, {36.986778, -86.449688},
{36.983132, -86.449418}, {36.9866555, -86.4512954}, {36.983477, -86.450575},
{36.9840776, -86.451948}, {36.988056, -86.4510331}, {36.9873294, -86.4534493},
{36.977864, -86.487877}, {36.9889555, -86.4527976}, {36.9861194, -86.4506813},
{36.9889791, -86.4526608}, {36.9866844, -86.4556855}, {36.971178, -86.459228},
{36.981803, -86.487619}, {36.965342, -86.448679}, {36.975815, -86.450953},
{36.987379, -86.464847}, {36.989625, -86.456275}, {36.964885, -86.485411},
{36.990075, -86.456833}, {36.990483, -86.457362}, {36.9654762, -86.486342},
{36.9659599, -86.4869695}, {36.981395, -86.466238}, {36.9671529, -86.4885724},
{36.964945, -86.485334}, {36.974597, -86.459917}, {36.976509, -86.461731},
{36.977004, -86.462559}, {36.970133, -86.455398}, {36.972564, -86.456998},
{36.972322, -86.452937}, {36.978104, -86.461491}, {36.9887412, -86.4578341},
{36.979823, -86.474978}, {36.980483, -86.474509}, {36.982183, -86.481},
{36.982609, -86.481618}, {36.99075, -86.47141}, {36.97491, -86.472684},
{36.976468, -86.470732}, {36.985399, -86.483739}, {36.985201, -86.483467},
{36.984982, -86.48318}, {36.9845164, -86.4825803}, {36.983842, -86.482267},
{36.98164, -86.483911}, {36.981253, -86.485548}, {36.981271, -86.486431},
{36.962159, -86.467904}, {36.981237, -86.475538}, {36.981592, -86.475209},
{36.980545, -86.474648}, {36.981129, -86.474187}, {36.980077, -86.473015},
{36.979931, -86.473142}, {36.965173, -86.46582}, {36.966676, -86.464868},
{36.967642, -86.464259}, {36.981944, -86.476398}, {36.969913, -86.462792},
{36.969981, -86.462747}, {36.970697, -86.462295}, {36.983504, -86.479294},
{36.983949, -86.479879}, {36.974301, -86.458998}, {36.960523, -86.462626},
{36.960601, -86.463164}, {36.976108, -86.457514}, {36.960806, -86.465359},
{36.960904, -86.466922}, {36.959584, -86.46869}, {36.960066, -86.469609},
{36.961737, -86.465245}, {36.958981, -86.462149}, {36.973579, -86.457935},
{36.964576, -86.454072}, {36.973175, -86.457}, {36.981623, -86.465225},
{36.964573, -86.447857}, {36.977334, -86.460063}, {36.978542, -86.459232},
{36.980881, -86.45242}, {36.982019, -86.451542}, {36.984371, -86.447947},
{36.959522, -86.471746}, {36.981821, -86.488202}, {36.981357, -86.488221},
{36.974575, -86.476016}, {36.981245, -86.488232}, {36.962977, -86.475635},
{36.972616, -86.455309}, {36.978575, -86.4548846}, {36.978925, -86.453713},
{36.971938, -86.450061}, {36.962691, -86.473652}, {36.977977, -86.486509},
{36.96243, -86.473355}, {36.979278, -86.451302}, {36.979513, -86.450036}};
```

In[ ]:=
```
edgew = {49.745000000000005, 53.882000000000005, 123.72, 37.59, 101.14, 348.46399999999994,
295.665, 252.184, 213.21199999999996, 469.839, 295.665, 195.117, 233.986, 13.837,
297.313, 140.959, 297.313, 138.575, 54.594, 6.49, 276.186, 366.878, 209.598,
114.662, 96.04199999999999, 250.968, 154.129, 72.78299999999999, 854.687, 308.172,
222.416, 136.07600000000002, 42.768, 103.366, 136.076, 127.126, 124.01400000000001,
```

```
103.366, 340.203, 124.01400000000001, 209.59799999999998, 56.528999999999996,
164.073, 87.00000000000001, 90.529, 14.252, 58.103, 94.234, 487.39099999999996,
34.387, 125.888, 124.94800000000001, 287.525, 323.669, 118.695, 154.224, 287.525,
169.435, 154.224, 287.55400000000003, 201.03599999999997, 75.279, 75.279, 84.017,
52.55900000000001, 52.559000000000005, 149.65099999999998, 11.641, 155.337,
195.11700000000002, 11.641, 194.288, 218.153, 162.624, 143.19899999999996,
110.19000000000001, 163.43099999999998, 480.15199999999993, 363.2829999999999,
212.77, 163.43099999999998, 102.241, 53.882000000000005, 53.944, 55.577, 111.297,
29.8, 83.59100000000001, 123.602, 29.8, 83.591, 111.472, 81.295, 27.631, 46.666,
82.202, 53.33, 167.56599999999997, 261.84, 90.61899999999999, 120.471, 115.303,
150.145, 120.471, 144.868, 535.959, 404.26700000000005, 324.53999999999996,
136.532, 144.86799999999997, 292.852, 305.028, 15.514, 116.67899999999999, 239.114,
15.514, 111.374, 227.08500000000004, 6.868, 89.05, 210.911, 111.37400000000001,
116.452, 191.47, 50.49, 89.05000000000001, 55.342999999999996, 197.92000000000004,
55.343, 84.052, 421.924, 168.89999999999998, 197.92, 129.54500000000002,
170.18499999999997, 26.994, 249.75600000000003, 177.522, 99.71, 249.75600000000003,
154.18399999999997, 160.33299999999997, 271.52699999999993, 99.71000000000001,
160.333, 271.527, 29.422000000000004, 29.421999999999997, 180.601, 202.424,
97.032, 98.106, 236.512, 363.67499999999995, 154.18399999999997, 32.945, 108.866,
120.871, 342.219, 32.945, 225.46, 290.992, 52.899, 59.24, 140.23999999999998, 86.38,
67.076, 108.793, 144.563, 79.84, 79.84, 50.489999999999995, 92.80300000000001,
31.919999999999998, 111.47200000000001, 45.184, 45.184, 99.48, 397.89099999999996,
37.59, 92.80300000000001, 31.92, 131.768, 484.3980000000001, 324.5399999999999,
167.82999999999998, 292.85200000000003, 172.158, 409.95200000000006, 484.39799999999985,
116.67899999999999, 136.53199999999998, 94.463, 246.70599999999996, 100.438,
169.43500000000003, 93.60199999999999, 165.73000000000002, 209.14200000000002,
305.27599999999995, 93.602, 85.568, 53.227000000000004, 16.768, 162.714, 323.669,
162.714, 90.529, 208.762, 309.97700000000003, 102.24100000000001, 593.7129999999999,
410.95899999999995, 148.84, 180.789, 81.656, 164.073, 81.656, 85.568, 55.129, 1185.553,
102.05600000000001, 186.164, 669.7740000000001, 58.087999999999994, 42.768, 298.112,
33.873000000000005, 17.585, 62.715, 296.71099999999996, 85.12700000000001, 17.585,
45.785, 126.815, 85.127, 98.298, 224.977, 114.968, 231.16299999999995, 149.595,
103.468, 149.595, 118.256, 275.52, 140.50799999999998, 104.409, 373.3610000000001,
461.29599999999994, 136.588, 73.003, 52.092, 70.353, 33.076, 73.003, 37.186, 33.076,
37.18599999999999, 89.131, 215.69499999999996, 114.66199999999999, 307.182, 6.502,
162.716, 140.64600000000002, 71.889, 133.41400000000002, 235.533, 72.55000000000001,
140.96200000000002, 249.70300000000003, 158.144, 140.96200000000002, 60.132999999999996,
53.227000000000004, 165.402, 296.71099999999996, 61.887, 195.577, 120.99300000000001,
118.346, 223.998, 351.21200000000005, 148.264, 47.367000000000004, 94.466, 221.603,
29.198999999999998, 89.131, 110.635, 221.603, 480.15200000000004, 110.635, 103.867,
363.2829999999999, 193.59699999999998, 35.93, 74.047, 70.02999999999999, 289.436,
63.257, 134.634, 55.12700000000001, 30.095000000000002, 7.031, 73.35199999999999,
72.712, 72.712, 103.11199999999998, 38.643, 9.659, 160.58100000000002, 38.637,
51.483, 23.98, 9.664, 63.904, 36.58, 20.054, 106.18400000000001, 143.612,
42.042, 45.419000000000004, 118.256, 421.61, 103.468, 214.05900000000003,
100.35600000000001, 144.706, 101.383, 297.28299999999996, 122.533, 287.55400000000003,
297.28299999999996, 223.998, 272.027, 227.08499999999998, 485.246, 129.54500000000002,
669.7739999999999, 680.748, 485.24600000000004, 369.843, 116.452, 71.582, 71.582,
98.29800000000002, 28.603, 37.498, 121.29499999999999, 342.451, 55.577, 59.516,
111.297, 150.09700000000004, 47.367000000000004, 53.869, 59.516000000000005, 66.021,
92.24000000000001, 42.248000000000005, 286.4940000000001, 191.46999999999997,
```

```
355.171, 239.11399999999998, 98.106, 42.248000000000005, 235.45399999999995,
348.46400000000006, 97.032, 97.74600000000001, 237.06599999999997, 380.342, 127.664,
97.74600000000001, 340.20300000000003, 397.891, 127.664, 114.08099999999999,
842.1770000000001, 380.434, 29.198999999999998, 52.092, 114.32499999999999, 39.538,
146.15, 150.145, 222.743, 103.867, 102.233, 380.4340000000001, 181.585, 138.5, 102.233,
184.788, 302.52200000000005, 209.142, 101.14, 96.703, 363.67499999999995, 184.788,
222.416, 97.743, 236.512, 96.47999999999999, 82.256, 229.08000000000004, 33.455,
177.01799999999997, 305.27599999999995, 186.164, 177.018, 28.98, 28.898, 7.704,
14.398, 96.47999999999999, 127.12600000000002, 237.06599999999997, 97.743, 51.904,
298.11199999999997, 235.454, 72.783, 376.5430000000001, 287.02, 151.183, 71.889,
222.743, 114.325, 167.13100000000003, 45.41899999999999, 62.787000000000006, 167.131,
52.657, 103.383, 104.423, 138.024, 103.38300000000001, 111.79, 37.452999999999996,
69.53899999999999, 69.53900000000002, 224.977, 138.024, 102.779, 296.613, 111.305,
151.433, 49.745000000000005, 100.59, 108.403, 111.305, 154.516, 52.657, 108.403,
141.04000000000002, 108.485, 112.608, 151.058, 154.51600000000002, 121.935,
244.75899999999996, 108.866, 252.184, 54.594, 154.53000000000003, 278.50800000000004,
231.42600000000002, 177.51300000000003, 129.499, 351.212, 120.993, 129.499, 359.076,
124.08699999999999, 31.123, 14.034, 16.768, 147.031, 277.353, 229.07999999999998,
14.034, 102.05600000000001, 144.87099999999998, 130.776, 71.285, 277.353, 82.256,
161.328, 71.285, 122.53300000000002, 168.731, 161.32799999999997, 62.787000000000006,
42.041999999999994, 168.731, 342.4509999999999, 187.352, 231.163, 191.34000000000003,
70.182, 41.614, 30.198000000000004, 70.182, 140.646, 64.328, 11.311, 64.902, 11.311,
205.65599999999998, 302.21399999999994, 136.588, 162.716, 152.90800000000002,
72.69300000000001, 152.90800000000002, 72.69300000000001, 96.79, 121.29499999999999,
126.50399999999999, 210.679, 82.436, 181.478, 136.542, 104.597, 262.145, 136.542,
162.624, 194.288, 273.13300000000004, 182.304, 136.796, 67.292, 593.7130000000001,
95.951, 108.793, 148.84, 125.78500000000001, 437.281, 287.02, 138.57500000000002,
282.63, 74.856, 67.29199999999999, 321.962, 110.10499999999999, 94.779,
211.08700000000002, 110.10499999999999, 854.6870000000004, 241.02599999999998,
146.361, 67.64800000000001, 241.02599999999995, 50.597, 165.402, 26.994, 108.173,
340.581, 49.243, 43.812, 49.243, 234.17599999999996, 135.443, 123.60199999999999,
251.66299999999998, 290.992, 282.271, 61.745000000000005, 251.66299999999995,
8.083, 61.745000000000005, 121.93499999999999, 8.083, 70.73, 123.64699999999999,
55.433, 70.73, 120.871, 52.544000000000004, 55.433, 26.2, 52.544000000000004,
563.806, 94.436, 26.2, 87.758, 94.436, 658.8639999999999, 33.730000000000004, 87.758,
113.90700000000001, 33.730000000000004, 138.5, 177.726, 113.90700000000001, 218.153,
120.048, 177.726, 103.506, 120.04799999999999, 212.77, 105.039, 103.506, 208.762,
34.031, 105.039, 309.97700000000003, 59.24, 86.72, 34.031, 124.225, 252.748, 124.225,
132.891, 181.478, 118.83, 252.748, 131.953, 125.78499999999998, 289.436, 102.779,
109.84599999999999, 40.169, 125.90899999999999, 112.608, 153.247, 151.433, 60.927,
108.485, 173.48499999999996, 104.423, 125.90899999999999, 193.85000000000002,
74.047, 193.85, 115.303, 151.183, 166.36, 166.35999999999999, 44.564, 33.612,
60.724, 125.026, 48.510000000000005, 86.72000000000001, 125.026, 44.564, 182.948,
43.126, 55.129, 210.679, 130.81, 130.81, 130.776, 158.144, 116.20899999999999,
44.487, 44.487, 117.638, 132.074, 95.74, 118.83, 134.927, 132.074, 47.201, 134.927,
262.14500000000004, 47.201, 90.569, 355.08600000000007, 117.01100000000002, 90.569,
233.97000000000003, 92.791, 117.011, 193.59699999999998, 115.09799999999998,
92.791, 249.703, 105.18599999999999, 82.436, 14.530999999999999, 105.186, 87.0,
14.530999999999999, 51.904, 95.332, 95.332, 421.92400000000004, 65.425, 54.981,
55.922, 65.425, 77.535, 77.535, 72.55000000000001, 82.587, 72.81800000000001,
37.722, 82.587, 140.50799999999998, 205.65600000000003, 206.79100000000003,
```

```
275.52, 206.791, 277.154, 261.84, 131.768, 234.17600000000004, 114.08099999999999,
54.292, 82.37, 54.292, 52.167, 82.37, 112.333, 165.73, 369.84299999999996,
305.028, 404.26700000000005, 409.95199999999994, 127.17600000000002, 112.333,
272.027, 124.08699999999999, 127.176, 118.34599999999999, 75.923, 75.923, 286.494,
210.95400000000004, 359.07599999999996, 148.26399999999998, 151.97799999999998,
210.954, 302.214, 154.53, 151.978, 758.4139999999999, 52.167, 53.86899999999999,
77.481, 67.113, 67.113, 77.481, 92.24000000000001, 66.021, 348.47800000000007,
56.529, 61.887, 348.47800000000007, 62.715, 58.088, 60.10900000000001, 60.109,
61.798, 205.485, 409.79900000000004, 156.427, 152.1, 205.485, 132.057, 278.508,
105.82, 143.199, 50.04, 105.82, 139.886, 58.103, 50.04, 140.24, 120.89100000000002,
221.19, 152.1, 133.41400000000002, 100.81800000000001, 221.19, 104.998, 120.939,
100.818, 277.15399999999994, 57.915, 120.939, 441.597, 283.77500000000003, 174.325,
106.964, 182.94799999999998, 70.35300000000001, 196.572, 106.964, 658.8639999999999,
233.98600000000005, 76.705, 76.70499999999998, 45.785, 283.775, 340.581, 421.61,
148.529, 196.572, 563.806, 40.649, 148.529, 342.219, 40.649, 85.408, 602.413,
282.271, 85.408, 244.75900000000001, 552.686, 225.46, 135.443, 552.686, 373.361,
437.28099999999995, 95.951, 109.84599999999999, 122.521, 214.059, 506.84599999999995,
53.944, 60.133, 94.46600000000001, 122.52099999999999, 196.29100000000003, 52.899,
139.886, 273.13300000000004, 110.19000000000001, 33.455, 61.79799999999999, 107.904,
258.8190000000001, 177.51299999999998, 562.441, 235.533, 376.22999999999996,
231.42600000000004, 156.42700000000002, 145.902, 376.23, 506.846, 112.069,
145.90200000000002, 54.493, 112.069, 51.526999999999994, 110.432, 153.935,
153.24699999999999, 301.04799999999994, 106.381, 144.563, 114.66, 146.15, 110.737,
110.43199999999999, 151.058, 81.373, 139.05100000000002, 110.73700000000002,
261.02099999999996, 183.483, 173.48500000000004, 139.05100000000002, 296.613, 40.169,
195.343, 39.538, 176.372, 237.99800000000002, 122.331, 96.79, 12.431000000000001,
370.336, 114.66, 67.076, 12.431000000000001, 195.34300000000002, 210.43199999999996,
6.49, 211.087, 275.80199999999996, 104.998, 71.576, 71.57600000000001, 51.862,
54.492999999999995, 104.597, 233.96999999999997, 116.209, 181.585, 70.458, 9.556,
168.89999999999998, 65.333, 70.458, 120.89099999999999, 43.812, 65.333, 77.465,
107.069, 100.438, 77.465, 94.23400000000001, 33.612, 213.21200000000002, 94.463,
246.70600000000005, 201.036, 9.556, 107.89000000000001, 118.695, 333.08299999999997,
360.984, 90.829, 92.005, 333.08299999999997, 360.9840000000001, 13.837, 92.005,
758.414, 31.123, 386.481, 132.057, 386.48100000000005, 69.712, 152.01600000000002,
131.953, 259.038, 117.638, 409.7989999999999, 282.63, 153.37900000000002,
140.959, 14.252, 94.779, 48.51, 84.48800000000001, 14.141, 84.488, 137.086,
72.744, 67.648, 50.596999999999994, 72.744, 123.647, 150.09699999999998, 245.147,
302.522, 842.1769999999999, 469.83900000000006, 245.147, 32.694, 81.295,
35.256, 32.693999999999996, 191.34000000000003, 46.666000000000004, 82.202,
35.256, 53.33, 81.576, 27.631, 290.28, 81.576, 187.352, 167.56600000000003,
155.76199999999997, 290.2799999999999, 195.57700000000003, 78.464, 155.762,
114.96799999999999, 126.815, 78.464, 461.29600000000005, 366.38900000000007,
34.387, 680.7479999999998, 100.35600000000001, 202.424, 277.69100000000003,
49.115, 37.722, 54.98100000000001, 49.115, 76.771, 14.141, 55.922, 76.771,
19.769, 19.769, 137.086, 155.33700000000002, 187.31, 104.40899999999999, 180.601,
120.272, 187.31, 355.17100000000005, 284.173, 120.272, 562.441, 6.502, 72.818,
602.4129999999999, 8.553, 284.173, 167.82999999999998, 89.169, 8.553, 172.158,
57.915, 89.169, 535.9590000000001, 30.198, 307.18199999999996, 146.36100000000002,
33.873000000000005, 90.619, 41.614, 90.82899999999998, 240.627, 441.597, 48.596,
196.50600000000003, 48.596, 108.173, 355.086, 240.627, 169.11299999999997, 139.316,
196.506, 104.233, 425.52699999999993, 233.86399999999998, 139.31600000000003,
```

```
101.383, 233.86399999999992, 425.5269999999999, 97.685, 182.304, 97.685,
277.69100000000003, 104.233, 144.706, 177.522, 28.603, 174.325, 94.484, 37.498,
84.017, 149.651, 196.29099999999997, 69.712, 94.484, 107.904, 215.69500000000002,
308.17199999999997, 60.724, 51.527, 154.03300000000002, 153.37900000000002,
250.96799999999996, 410.95899999999995, 154.03300000000002, 376.54300000000006,
96.042, 37.453, 60.927, 183.483, 111.78999999999999, 261.021, 70.03, 153.935,
136.79600000000002, 76.498, 76.498, 51.67, 103.112, 51.861999999999995, 12.492,
51.67000000000001, 124.948, 96.703, 380.342, 99.48, 12.492, 125.888, 179.866,
170.18499999999997, 95.74000000000001, 152.016, 258.81899999999996, 111.307,
53.667, 6.868, 115.098, 111.307, 259.038, 132.891, 126.50399999999999, 39.29,
179.86599999999999, 84.05199999999999, 122.331, 366.38900000000007, 1185.553,
39.290000000000006, 141.04000000000002, 115.477, 210.911, 100.59, 123.72, 115.477};
```

Initialization

```
In[•]:= SetDirectory[NotebookDirectory[]];
        设置目录        当前笔记本的目录
        locations = {};
        Off[SetDelayed::shape]
        ...  设置延迟
        Off[Set::shape]
        ...  赋值
        g = GeoGraphics[];
            地理图形
        graph = Graph[Import["data\\walmart_tacobell.graphml"],
                图     导入
           EdgeWeight → edgew, VertexCoordinates → vertexcoords];
           边的权值              顶点坐标
```

Starting Window

```
In[ ]:= startingMenu[] :=
  CreateDialog[
  创建对话框
   {Graphics[{
    图形
      {Inset[ImageResize[Import["Images/PixelArtWorld.png"], {900, 900}],
       插图   调整图像大小   导入
        {Center, Center}, {Center, Center}, {900, 900}]},
         居中     居中       居中     居中
      {Text[Style["Mapematica", 50, FontFamily → "Comic Sans MS", Red], {Center, 825}]},
       文本  样式                    字体系列                        红色    居中
      (*This text displays of the app*)
      {Inset[Button[Style["Add Locations", 20],
       插图   按钮    样式
         locationSelect[];
         DialogReturn[], BaseStyle → {15}, ImageSize → {300, 150}], {250, 600}]},
         对话框返回        基本样式            图像尺寸
      {Inset[Button[Style["Navigate", 20], adjustPalette[]; DialogReturn[],
       插图   按钮    样式                                   对话框返回
         BaseStyle → {15}, ImageSize → {300, 150}], {650, 600}]},
         基本样式            图像尺寸
      {Inset[Button[Style["Quit", 20], DialogReturn[];
       插图   按钮    样式    退出内核     对话框返回
         locations = {}, BaseStyle → {15}, ImageSize → {300, 150}], {Center, 350}]}
                         基本样式            图像尺寸                      居中
      (*This button will exit the window*)
     }, ImageSize → {900, 900}, PlotRange → {{0, 900}, {0, 900}}]},
       图像尺寸                  绘制范围
  Background → Gray, WindowSize → {900, 900},
  背景色         灰色   视窗大小
  WindowTitle → "Mapematica", WindowMargins → Automatic, Saveable → False
  视窗标题                    视窗边幅          自动        可保存否    假
 ]
```

Add Locations To Drive

```
In[ ]:= locationSelect[] := DynamicModule[{},
                            动态模块
    CreateDialog[
    创建对话框
     {Graphics[{
      图形
        {Inset[ImageResize[Import["Images/PixelArtWorld.png"], {900, 900}],
         插图   调整图像大小   导入
          {Center, Center}, {Center, Center}, {900, 900}]},
           居中     居中       居中     居中
        {Text[Style["Add Locations", 50, FontFamily → "Comic Sans MS", Red], {Center, 825}]},
         文本  样式                      字体系列                      红色    居中
        (*This text displays the title if the pane*)
        {Inset[Button[Style["Use Address", 20],
         插图   按钮    样式
           addressPalette[];
           DialogReturn[], BaseStyle → {15}, ImageSize → {300, 150}], {250, 600}]},
           对话框返回        基本样式          图像尺寸
        {Inset[Button[Style["Use Map", 20], DialogReturn[];
         插图   按钮    样式        映射       对话框返回
           mapPalette[]; , BaseStyle → {15}, ImageSize → {300, 150}], {600, 600}]} ,
                           基本样式          图像尺寸
        {Inset[Button[Style["Return", 20], startingMenu[]; DialogReturn[],
         插图   按钮    样式    返回                          对话框返回
           BaseStyle → {15}, ImageSize → {300, 150}], {Center, 350}]}
           基本样式          图像尺寸                    居中
      }, ImageSize → {900, 900}, PlotRange → {{0, 900}, {0, 900}}]},
         图像尺寸                 绘制范围
     Background → Gray, WindowSize → {900, 900},
     背景色        灰色   视窗大小
     WindowTitle → "Mapematica", WindowMargins → Automatic, Saveable → False
     视窗标题                    视窗边幅          自动        可保存否    假
    ]
   ]

In[ ]:= addressPalette[] := CreateDialog[DynamicModule[{loc = "", coords, valid = True, time = ""},
                            创建对话框     动态模块                                    真
     Dynamic@Graphics[{
     动态     图形
        {Text[Style["Add By Address", 50, FontFamily → "Comic Sans MS"], {Center, 825}]},
         文本  样式                      字体系列                      居中
        {Text[Style["Enter Address Here:", 25, FontFamily → "Comic Sans MS"], {180, 500}]},
         文本  样式                  当前位置     字体系列
        {Inset[InputField[Dynamic@loc, String, FieldSize → Large], {180, Center}]},
         插图   输入栏      动态          字符串   输入栏大小   大            居中
        {Inset[InputField[Dynamic@time, String, FieldSize → Large], {180, 550}]},
         插图   输入栏      动态          字符串   输入栏大小   大
        {Text[Style["Enter Time Here:", 25, FontFamily → "Comic Sans MS"], {180, 600}]},
         文本  样式                  当前位置     字体系列
```

```
        文本 样式                    绘制位置      字体系列
      {Inset[Button[Style["Add", 15], coords = Interpreter["StreetAddress"][loc];
       插图  按钮   样式                          解释器
         If[! FailureQ[coords] && loc ≠ "" && time ≠ "",
         如果 失败判定
          AppendTo[locations, {coords[[1]], loc, ToExpression@time}];
          附加                                   转换为表达式
          addressPalette[];
          DialogReturn[], valid = False],
          对话框返回              假
         BaseStyle → {15}, ImageSize → {75, 30}], {500, 300}]},
         基本样式          图像尺寸
      {Inset[Button[Style["Return", 15], locationSelect[];
       插图  按钮   样式    返回
         DialogReturn[], BaseStyle → {15}, ImageSize → {75, 30}], {400, 300}]},
         对话框返回       基本样式          图像尺寸
      {If[! valid,
       如果
        {Text[Style["This Address is Invalid or Your Submission is Incomplete",
         文本 样式
           30, FontFamily → "Comic Sans MS", Red], {Center, 250}]}]},
              字体系列                      红色   居中
      {Text[Style["How to Add by Address:\n\nType in the dialog box the address
       文本 样式
           of\n your desired destination\n\nIf your address is invalid or
                                          如果
           does not exist, you will\n not be able to add it\n\nRelative
           locations are accepted\n but are not guaranteed to be accurate!",
         20, FontFamily → "Comic Sans MS", TextAlignment → Center], {600, 550}]}
             字体系列                     文本对齐        居中
    },
    ImageSize → {900, 900}, PlotRange → {{0, 900}, {0, 900}}]
    图像尺寸                 绘制范围
  ],
  Background → LightBlue, WindowSize → {900, 900}, WindowClickSelect → True,
  背景色       浅蓝色      视窗大小                 视窗点击选择          真
  WindowFloating → False, WindowMargins → Automatic, Saveable → False, WindowTitle → "Map"
  视窗漂浮          假     视窗边幅          自动       可保存否    假     视窗标题       映射
 ]

In[ ]:= mapPalette[] :=
  CreateDialog[DynamicModule[{addLoc = False, pos, warn = False, name = False, marker = {}},
  创建对话框   动态模块                假                假              假
    Dynamic@Graphics[{
    动态     图形
      {If[addLoc, Text[Style["Click On Map to Add\nNew Location",
       如果       文本 样式          … 映射
          35, Red, FontFamily → "Comic Sans MS"], {Center, 825}],
             红色 字体系列                          居中
        Text[Style["Add By Map", 50, FontFamily → "Comic Sans MS"], {Center, 850}]]},
        文本 样式          映射      字体系列                          居中
```

```
      {Inset[Button[Style["Return", 15], locationSelect[];
      插图  按钮  样式  返回
        DialogReturn[], BaseStyle → {15}, ImageSize → {75, 50}], {80, 275}]},
        对话框返回  基本样式  图像尺寸
      {Inset[Button[Style["Add Location", 15], pos = addPalette[];
      插图  按钮  样式
        marker = GeoMarker@pos;
                 地理标记
        name = True, BaseStyle → {15},
               真  基本样式
        ImageSize → {125, 50}, Method → "Queued"], {205, 275}]},
        图像尺寸  方法
      {Inset[Button[Style["Name Select", 15], If[name, mapCheck[pos];
      插图  按钮  样式  选择  如果
         warn = False, warn = True],
                假  真
        BaseStyle → {15}, ImageSize → {200, 50}], {Center, 275}]},
        基本样式  图像尺寸  居中
      {If[warn, Text[Style["Select a Location\nBefore Proceding", 35,
      如果  文本  样式  选择  在之前
         Red, FontFamily → "Comic Sans MS"], {700, 350}]]},
         红色  字体系列
      {Text[Style["How To Use Map:\n\nChoose add location to open the
      文本  样式  映射
          selection map\n\n Click on map to change the location
          you are adding\n\n choose \"Select Location\" when you
                                    选择
          are ready\n\nFinally, continue to \"Name Select\"", 20,
                                                 选择
        FontFamily → "Comic Sans MS", TextAlignment → Center, Red], {650, 600}]},
        字体系列  文本对齐  居中  红色
      {Inset[GeoGraphics[marker, ImageSize → Medium, GeoRange → Quantity[1, "Miles"]],
      插图  地理图形  图像尺寸  中  地理范围  数量
       {200, 550}]}
    },
    ImageSize → {900, 900}, PlotRange → {{0, 900}, {0, 900}}]
    图像尺寸  绘制范围
 ],
 Background → LightBlue, WindowSize → {900, 900}, WindowClickSelect → True,
 背景色  浅蓝色  视窗大小  视窗点击选择  真
 WindowFloating → False, WindowMargins → Automatic, Saveable → False, WindowTitle → "Map"
 视窗漂浮  假  视窗边幅  自动  可保存否  假  视窗标题  映射
]
```

```
In[•]:= addPalette[] := DialogInput[DynamicModule[
                        对话框输入    动态模块
    {map = DynamicGeoGraphics[ImageSize → 200], pos, mousePos, marker, marked = False},
          动态地理图形          图像尺寸                                          假
    Column[{
    列
      EventHandler[Dynamic@map,
      事件句柄      动态
       {"MouseClicked" :→ (mousePos = MousePosition["GraphicsAbsolute"];
                                     鼠标位置
          pos = MousePosition["Graphics"];
                鼠标位置          图形
          If[mousePos[[1]] < 570,
          如果
           marker = {N[trans[pos][[1, 1, 1, 2]], 5], N[trans[pos][[1, 1, 2, 2]], 5]};
                     数值运算                          数值运算
           marker[[1]] = ToExpression@ToString@marker[[1]];
                       转换为表达式    转换为字符串
           marker[[2]] = ToExpression@ToString@marker[[2]];
                       转换为表达式    转换为字符串
           map = DynamicGeoGraphics[GeoMarker[GeoPosition[marker]], ImageSize → 200];
                 动态地理图形          地理标记    测地位置                 图像尺寸
           marked = True
                    真
          ]
         )}, PassEventsDown → True],
             向内传递事件       真
      Row[{
      行
        Spacer[10],
        点间隔
        Button[Style["Return", 15],
        按钮    样式    返回
         If[marked, DialogReturn[marker]], BaseStyle → {15}, ImageSize → {75, 50}]
         如果       对话框返回                基本样式          图像尺寸
       }, Editable → False]
          可编辑      假
     }, Editable → False]
        可编辑      假
  ], Background → LightBlue, WindowSize → {625, 700}, WindowClickSelect → True,
     背景色        浅蓝色      视窗大小                  视窗点击选择          真
  WindowFloating → False, WindowMargins → Automatic, Saveable → False, WindowTitle → "Map"
  视窗漂浮          假      视窗边幅          自动       可保存否     假     视窗标题        映射
 ]
```

```
In[•]:= mapCheck[pos_] := DynamicModule[{name = "", time = ""},
                          动态模块
    CreateDialog[
    创建对话框
     Graphics[{
     图形
       {Text[Style["Confirm Location", 20, FontFamily → "Comic Sans MS"], {Center, 275}]},
        文本  样式                          字体系列                          居中
       {Inset[InputField[Dynamic@name, String, FieldSize → {22, 1.1}], {260, 225}]},
        插图   输入栏       动态            字符串   输入栏大小
       {Inset[InputField[Dynamic@time, String, FieldSize → {22, 1.1}], {260, 195}]},
        插图   输入栏       动态            字符串   输入栏大小
       {Text[Style["Required Time:", 15, FontFamily → "Comic Sans MS"], {80, 197}]},
        文本  样式                          字体系列
       {Inset[Button[Style["OK", 15], DialogReturn[];
        插图   按钮    样式              对话框返回
          AppendTo[locations, {pos, name, ToExpression@time}]
          附加                            转换为表达式
          , BaseStyle → {15}, ImageSize → {50, 30}], {350, 40}]},
            基本样式            图像尺寸
       {Inset[Button[Style["Cancel", 15],
        插图   按钮    样式    约简
          DialogReturn[], BaseStyle → {15}, ImageSize → {80, 30}], {260, 40}]},
          对话框返回        基本样式            图像尺寸
       {Text[Style["Name of Location:", 15, FontFamily → "Comic Sans MS"], {80, 225}]},
        文本  样式                          字体系列
       {Text[Style["Directions:", 15, FontFamily → "Comic Sans MS", Red], {55, 125}]},
        文本  样式                      字体系列                        红色
       {Text[Style["Name the Location You are Adding Above",
        文本  样式                                          上
          15, FontFamily → "Comic Sans MS", Red], {160, 100}]},
              字体系列                        红色
       {Text[Style["Then Click OK to Save Your Location", 15,
        文本  样式                      保存
          FontFamily → "Comic Sans MS", Red], {143.5, 80}]}
          字体系列                        红色
      }, ImageSize → {400, 300}, PlotRange → {{0, 400}, {0, 300}}],
         图像尺寸                  绘制范围
     Background → LightBlue, WindowSize → {400, 300},
     背景色         浅蓝色       视窗大小
     WindowTitle → "Confirm", WindowClickSelect → True,
     视窗标题                  视窗点击选择          真
     WindowFloating → False, WindowMargins → Automatic, Saveable → False
     视窗漂浮          假     视窗边幅          自动        可保存否     假
    ]
   ]
```

```
In[•]:= tr = "DisplayFunction" /. (CoordinatesToolOptions /. AbsoluteOptions[g]);
        显示函数                  坐标工具选项                 绝对设置
        trans[u_] := tr[u];
        trans[Automatic] = tr[Mean /@ (PlotRange /. AbsoluteOptions[g])];
                 自动          平均值   绘制范围      绝对设置
```

Navigate with Current Locations

```
In[•]:=
navigationSetup[] :=
 Module[{closeCoords, closeNodeId, locs, times, i, path = {}, graphics = {}, possible},
 模块
  (*sort locations by time smallest→largest*)
  locations = Sort[locations, #1[[3]] < #2[[3]] &];
              排序
  locs = locations[[All, 1]];
                   全部
  (*take user input and convert them to the nodes they closest correspond to*)
  {closeCoords, closeNodeId} = closestNodes[locs];
  (*find the shortest path to each destination*)
  For[i = 2, i ≤ Length@closeNodeId, i++,
  For循环        长度
   AppendTo[path, dijkstra[graph, closeNodeId[[i - 1]], closeNodeId[[i]]]]
   附加
  ];
  path = outputToCoords[path];
  For[i = 1, i ≤ Length@path, i++,
  For循环        长度
   AppendTo[graphics,
   附加
    {Style[Arrow[TravelDirections[path[[i]]]], Hue[i * .263], Thickness[0.003]]}];
     样式   箭头   行进方向                      色相          粗细
   AppendTo[graphics, {If[i == 1, Green, Black], PointSize[Large], Point[path[[i, 1]]]}];
   附加                 如果      绿色   黑色     点的大小    大       点
   AppendTo[graphics,
   附加
    {If[i == Length@path, Red, Black], PointSize[Large], Point[path[[i, Length@path[[i]]]]]}]
     如果   长度            红色  黑色     点的大小    大       点              长度
  ];
  possible = isPossible[];
  navigationPalette[possible, graphics]
 ]
```

In[ ]:=
```
navigationPalette[possible_, path_] :=
 CreateDialog[
 创建对话框
  {
   Dynamic@Graphics[{
   动态      图形
      {Text[Style[possible, 50, FontFamily → "Comic Sans MS", Red], {Center, 850}]},
       文本  样式                 字体系列                        红色    居中
      {Inset[Button[Style["Return", 15], adjustPalette[];
       插图   按钮    样式    返回
         DialogReturn[], BaseStyle → {15}, ImageSize → {75, 50}], {80, 275}]},
         对话框返回        基本样式            图像尺寸
      {Inset[GeoGraphics[path, ImageSize → 400], {550, 550}]},
       插图   地理图形            图像尺寸
      {Red, Disk[{50, 700}, 10]},
       红色  圆盘
      {Text[
       文本
        Style["- End of the trip", 25, FontFamily → "Comic Sans MS", Red], {170, 700}]},
        样式      结束                  字体系列                        红色
      {Green, Disk[{50, 600}, 10]},
       绿色    圆盘
      {Text[Style["- Start of the trip",
       文本  样式
         25, FontFamily → "Comic Sans MS", Green], {180, 600}]},
             字体系列                       绿色
      {Black, Disk[{50, 500}, 10]},
       黑色    圆盘
      {Text[Style["- Intermediate Stop", 25, FontFamily → "Comic Sans MS"], {190, 500}]}
       文本  样式                              字体系列
     }, ImageSize → {900, 900}, PlotRange → {{0, 900}, {0, 900}}]
        图像尺寸                  绘制范围
  },
  Background → LightBlue, WindowSize → {900, 900},
  背景色         浅蓝色       视窗大小
  WindowTitle → "Navigation", WindowMargins → Automatic, Saveable → False
  视窗标题                     视窗边幅          自动        可保存否    假
 ]
```

```
In[•]:= isPossible[] := Module[{time, tt, i, tripTime = 0, maxTime},
                        模块
          For[i = 2, i ≤ Length@locations, i++,
          For循环          长度
           maxTime = locations[[i, 3]];
           tt = TravelTime[{GeoPosition[locations[[i - 1, 1]]],
                行进时间        测地位置
               GeoPosition[locations[[i, 1]]]}, TravelMethod → "Driving"];
               测地位置                          行进方式
           time = tt[[1, 1, 1]];
           tripTime += time;
           If[tripTime > maxTime, Return["This Trip is Not Possible"]]
           如果                    返回                    逻辑非
          ];
          Return["This Trip is Possible"]
          返回
         ];

In[•]:= adjustPalette[] := CreateDialog[DynamicModule[{index = 1},
                           创建对话框      动态模块
          Dynamic@Graphics[{
          动态      图形
             {If[Length@locations < 2, Text[Style["Add More Destinations Before Navigating!",
              … 长度                    文本  样式                                在之前
                 35, Red, FontFamily → "Comic Sans MS"], {Center, 850}], Text[Style[
                     红色 字体系列                          居中            文本  样式
                 "Current Destinations", 50, FontFamily → "Comic Sans MS"], {Center, 850}]]},
                                             字体系列                          居中
             {Inset[Button[Style["Return", 15], startingMenu[];
              插图  按钮    样式     返回
                DialogReturn[], BaseStyle → {15}, ImageSize → {75, 50}], {80, 275}]},
                对话框返回        基本样式            图像尺寸
             {If[Length@locations ≥ 2, Inset[Button[Style["Navigate", 15],
              … 长度                    插图  按钮    样式
                 pleaseWait[];
                 DialogReturn[];
                 对话框返回
                 , Method → "Queued", BaseStyle → {15}, ImageSize → {75, 50}], {820, 275}]]},
                   方法                基本样式            图像尺寸
             {Inset[Button[Style["Previous", 15], index = newIndex[index, True],
              插图  按钮    样式                                                真
                BaseStyle → {15}, ImageSize → {75, 50}], {350, 350}]},
                基本样式            图像尺寸
             {Inset[Button[Style["Next", 15], index = newIndex[index, False],
              插图  按钮    样式                                          假
                BaseStyle → {15}, ImageSize → {75, 50}], {550, 350}]},
                基本样式            图像尺寸
             {If[Length@locations ≥ 1, Inset[GeoGraphics[GeoMarker[locations[[index, 1]]],
              … 长度                    插图  地理图形      地理标记
                 ImageSize → Medium, GeoRange → Quantity[1, "Miles"]], {Center, 600}]]},
                 图像尺寸            中          地理范围      数量                    居中
```

```
        {If[Length@locations ≥ 1, Inset[Button[Style["Remove Location", 15],
         ⋯ 长度                    插图  按钮   样式    去除
            locations = Drop[locations, index];
                        去掉元素
            index = 1, BaseStyle → {15}, ImageSize → {150, 50}], {Center, 275}]]},
                        基本样式          图像尺寸                  居中
        {If[Length@locations ≥ 1, Text[
         ⋯ 长度                    文本
           Style[locations[[index, 2]], 20, FontFamily → "Comic Sans MS"], {Center, 800}]]}
           样式                             字体系列                          居中
       },
       ImageSize → {900, 900}, PlotRange → {{0, 900}, {0, 900}}]
       图像尺寸                  绘制范围
    ],
    Background → LightBlue, WindowSize → {900, 900}, WindowClickSelect → True,
    背景色        浅蓝色       视窗大小                 视窗点击选择          真
    WindowFloating → False, WindowMargins → Automatic, Saveable → False, WindowTitle → "Check"
    视窗漂浮          假      视窗边幅          自动        可保存否     假      视窗标题        校验
   ]

In[ ]:= newIndex[current_, previousQ_] := Module[{res = current},
                                          模块
    If[current == Length@locations && ! previousQ, Return[1]];
    如果        长度                                 返回
    If[current == 1 && previousQ, Return[Length@locations]];
    如果                          返回    长度
    If[previousQ, Return[res - 1], Return[res + 1]]
    如果          返回              返回
   ]
```

```
In[•]:= pleaseWait[] :=
  CreateDialog[
  创建对话框
   DynamicModule[{notClicked = True},
   动态模块                          真
    Dynamic@Graphics[{
    动态      图形
       {If[notClicked, Text[Style["Click Start to\nBegin Navigation",
        如果            文本  样式                      开始
           Green, 30, FontFamily → "Comic Sans MS"], {Center, 250}],
           绿色        字体系列                          居中
         Text[Style["Please Wait:\nCalculating Your Route!", Green, 30,
         文本  样式                                          绿色
           FontFamily → "Comic Sans MS"], {Center, 250}]]},
           字体系列                          居中
       {If[notClicked, Inset[Button[Style["START", 15], notClicked = False;
        如果            插图   按钮    样式                          假
           navigationSetup[];
           DialogReturn[], Method → "Queued", BaseStyle → {15}, ImageSize → {75, 50}]],
           对话框返回        方法                基本样式            图像尺寸

         Inset[ GATTON ACADEMY , {Center, 100}]]}
         插图                   居中

      }, ImageSize → {400, 300}, PlotRange → {{0, 400}, {0, 300}}]
         图像尺寸                 绘制范围
   ],
   Background → LightBlue, WindowSize → {400, 300},
   背景色        浅蓝色        视窗大小
   WindowTitle → "Please Wait", WindowClickSelect → True,
   视窗标题                     视窗点击选择          真
   WindowFloating → False, WindowMargins → Automatic, Saveable → False
   视窗漂浮          假      视窗边幅          自动        可保存否    假
  ]
```

```
In[•]:= startingMenu[]

Out[•]= NotebookObject[ Mapematica ]
```

Graph Management

```
In[•]:= (* Conducts Dijkstra's algorithm on a given graph "g" to find the shortest
  path from vertex "start" to vertex "finish". We assume that the vertices
  are labeled from {1,2,...,n}. The output is a list of three objects in
  order: the shortest path (as a list of the vertices visited, in order),
```

```
a list specifying the step on which each vertex was first opened,
and a list specifying the step on which each vertex was first closed. *)
dijkstra[g_, start_, finish_] :=
 Module[{vl = VertexList[g], el = EdgeList[g], wl = PropertyValue[g, EdgeWeight],
 模块          顶点列表              边的列表            属性值              边的权值
   path = {}, q, dist, prev, alt, done = False, u, v, nbh},
                                         假
  (* vl = vertex list
     el = edge list
      wl = edge weight list
       q = set of vertices yet to be processed
        dist = vector of shortest known distances from start to each vertex
         prev = vector of predecessors along the shortest path from start to each vertex
          alt = alternative distance to compare
           done = indicates whether or not the finish vertex has closed
           u,v = labels of vertices currently under examination
     nbd = neighborhood of vertex currently under examination
      i = iteration *)
  (* initialization *)
  q = vl;
  Do[
  Do循环
   dist[(vl[[k]])] = prev[(vl[[k]])] = Infinity,
                                       无穷大
   {k, 1, Length[vl]}];
          长度
  dist[start] = 0;
  prev[start] = start;
  (* main loop *)
  While[done == False,
  While循环      假
   u = Flatten[MinimalBy[q, dist]][[1]];
       压平      符合条件的最小值
   (* pick u as the vertex with the smallest known distance *)
   q = DeleteCases[q, u]; (* remove u from q *)
       删除匹配元素
   If[u == finish,
   如果
    done = True, (* if the finish vertex has just closed, terminate *)
           真
    nbh = Intersection[q, DeleteCases[VertexList[NeighborhoodGraph[g, u]], u]];
          交集            删除匹配元素  顶点列表      邻域图
    (* neighborhood of u *)
    Do[
    Do循环
     v = nbh[[k]];
     (* calculate the distance to v from u *)
     If[MemberQ[el, u ↔ v] == True,
     ⋯ 成员判定                真
      alt = dist[u] + wl[[(Flatten[Position[el, u ↔ v]][[1]])]]
                          压平      位置
```

```
        ];
        If[alt < dist[v],
        如果
         dist[v] = alt;
         prev[v] = u;
        ],
        {k, 1, Length[nbh]}
               长度
      ];
    ];
  ];
  (* back-tracing to get the path *)
  u = finish;
  While[u ≠ start,
  While循环
   path = Prepend[path, u];
          加在前面
   u = prev[u];
  ];
  path = Prepend[path, start];
         加在前面
  path
 ]
```

In[ ]:=
```
closestNodes[coords_] := Module[{nodes = {}, close = {}, i},
                                 模块
  For[i = 1, i ≤ Length@coords, i++,
  For循环        长度
   AppendTo[close, Nearest[vertexcoords, coords[[i]]]]
   附加              最接近
  ];
  close = Flatten@close;
          压平
  close = Partition[close, 2];
          划分
  For[i = 1, i ≤ Length@close, i++,
  For循环        长度
   AppendTo[nodes, VertexList[graph][[Flatten[Position[vertexcoords, close[[i]]]][[1]]]]]
   附加              顶点列表              压平      位置
  ];
  Return[{close, nodes}];
  返回
 ]
```

In[ ]:=
```
outputToCoords[nodes_] := Module[{coords = {}, coords2 = {}, i, j},
                          模块
  For[i = 1, i ≤ Length@nodes, i++,
  For循环        长度
   coords = {};
   For[j = 1, j ≤ Length@nodes[[i]], j++,
   For循环        长度
    AppendTo[coords,
    附加
     GeoPosition[vertexcoords[[Flatten[Position[VertexList[graph], nodes[[i, j]]]][[1]]]]]
     测地位置                    压平      位置       顶点列表
   ];
   AppendTo[coords2, coords];
   附加
  ];
  Return[coords2]
  返回
 ]
```