Assignment 2a - Island Models, Fitness Sharing, and Crowding

Joel Doumit
College of Computer Science
University of Idaho
Moscow, ID 83843
Email: doum6708@vandals.uidaho.edu

1. Output

1.1. Migration Proof

```
Migrants from population 1:
[[array([-1.39481912, -2.6483627 , -0.42768623, -3.05700464, -3.04080583,
         2.62577437, -1.28763136, -1.0872623 , -0.63641004, -1.57137948,
         2.11603021, -2.11336642,  1.07784975, -0.94914293,  0.26257943,
         1.58203759,  1.40847279, -1.59325385, -2.00324457,  0.73605448,
        -0.99212067, -2.57355289,  3.64529579,  3.19870311, -1.71335832,
         1.13967792, -3.15854406,  1.54651604, -0.26003285,
         1.89541277]), 114.38860010290168],
         [array([-1.39481912, -2.6483627 , -0.42768623,
         -3.05700464, -3.04080583,
         2.62577437,  0.21769552, -1.0872623 ,  1.05394345,
         -1.57137948,
         1.1950884 , -0.96712176, -2.5645344 ,  1.90845498,
         1.38427908,
         1.57705242, -2.33757771, -0.84945263,  0.51133036,
         -0.56222113,
         2.50331093,  2.52422838, -0.0524182 , -3.60118567,
         -1.71335832,
         0.84361462, -3.15854406, -1.75160862, -0.77667173,
         1.89541277]), 109.684806532334991]]
Population 2:
[[array([-1.38546159,  1.32000713, -2.75744711, -0.15869173,
-1.59629094,
         1.06602027,  3.53134903, -0.41559175,  0.77467106,
         0.29152919,
        -2.1353933 , -1.7162483 , -2.72989994, -1.75194542,
        0.96099567,
         1.55535332, -2.7486124 , -0.32661542, -2.44457103,
         3.04061778,
         2.15507329,  0.56305097, -0.81195275, -0.03057534,
         1.23870602,
         0.56883255,  0.39614537,  0.82152941,  0.23927194,
         1.26814193]), 82.53276798913804], [array([ 2.04751799,
         1.32000713, -0.9619002 ,  0.96925601, -1.59629094,
         0.66572924,  1.30201956, -0.41559175,  2.1741441 ,
         -1.47655807,
```

```
 0.88471255,  1.00974153, -2.72989994,  2.16458677,
 -3.40430148,
 1.55535332, -0.22183232,  1.10804487, -0.50204193,
 -0.84997895,
 2.15507329,  0.56305097, -0.81195275, -0.03057534,
 1.23870602,
-1.69318381,  0.39614537,  3.70641647,  0.45027592,
1.89198196]), 77.46548729098014], [array([-1.38546159,
1.32000713, -2.75744711, -0.15869173, -1.59629094,
 1.06602027,  3.53134903, -0.41559175,  0.77467106,
 3.08384093,
-2.1353933 , -1.7162483 , -2.72989994, -1.75194542,
0.96099567,
 1.55535332, -2.7486124 ,  0.3154596 , -0.50204193,
 -4.81592107,
 2.15507329,  0.56305097,  0.42944822,  2.64930339,
 1.23870602,
-2.91778842,  0.39614537, -1.58041866,  0.45027592,
1.26814193]), 116.8758093893934], [array([ 0.01240573,
1.32000713,  1.40355496, -0.15869173, -1.59629094,
 2.94677049, -1.53350581, -0.41559175,  0.77467106,
 0.29152919,
-2.1353933 ,  3.13740329, -2.72989994, -1.75194542,
0.96099567,
 1.55535332, -2.7486124 ,  0.3154596 , -2.78842285,
 -0.84997895,
 0.65687968,  0.56305097, -0.81195275, -0.03057534,
 1.23870602,
 0.56883255, -1.54791485,  0.82152941, -4.02195093,
 1.26814193]), 86.72034930336125], [array([-1.13061715,
 1.32000713, -2.75744711, -0.15869173, -1.59629094,
 1.06602027,  3.53134903, -0.41559175,  0.77467106,  0.29152919,
 2.18867712, -1.7162483 , -2.72989994, -1.75194542, -1.39576031,
 1.55535332, -2.7486124 ,  0.3154596 , -0.50204193, -0.84997895,
 2.15507329,  0.56305097, -0.81195275, -0.03057534,  0.28228095,
-2.91778842,  0.39614537, -1.58041866,  0.45027592,
1.26814193]), 77.5961785448705], [array([ 0.01240573,
1.32000713,  1.40355496, -0.15869173, -1.59629094,
 3.52642144, -1.53350581, -0.41559175,  0.77467106,  0.29152919,
-2.1353933 ,  1.1813175 , -2.72989994, -1.75194542,  0.96099567,
-1.97378918, -1.03569499,  0.3154596 , -2.78842285, -0.84997895,
 0.65687968,  0.56305097, -0.81195275, -0.03057534,  1.23870602,
 0.65976158,  0.39614537,  0.68013372,  0.45027592,
 1.89198196]), 60.67761434732863], [array([-3.21524486,
 1.32000713,  1.40355496, -0.15869173, -1.59629094,
-3.46397073, -1.53350581, -0.41559175,  2.1741441 , -1.47655807,
-1.55833703, -0.27659769, -2.72989994,  2.16458677,  1.18024501,
 4.62099273, -0.22183232,  0.62480884, -0.50204193, -0.84997895,
 0.12709395,  0.56305097, -0.81195275,  0.60073597,  1.23870602,
 2.71104509,  0.39614537,  1.40328852,  0.45027592,
 1.89198196]), 93.00343929112026], [array([ 1.39582793,
 -0.26542814,  1.1380319 , -1.02091322, -1.59629094,
-2.40253624,  1.52345431, -0.41559175, -0.81586845,  0.29152919,
 2.77927648,  3.13740329, -2.72989994, -1.75194542, -4.26789858,
 1.55535332, -2.7486124 ,  0.3154596 , -2.78842285, -0.84997895,
 0.65687968,  0.56305097, -0.81195275, -0.03057534,  1.23870602,
-2.91778842,  0.39614537, -1.58041866,  0.45027592,
1.26814193]), 96.7185183091327], [array([-1.39481912,
```

```
       -2.6483627 , -0.42768623, -3.05700464, -3.04080583,
        2.62577437, -1.28763136, -1.0872623 , -0.63641004, -1.57137948,
        2.11603021, -2.11336642,  1.07784975, -0.94914293,  0.26257943,
        1.58203759,  1.40847279, -1.59325385, -2.00324457,  0.73605448,
       -0.99212067, -2.57355289,  3.64529579,  3.19870311, -1.71335832,
        1.13967792, -3.15854406,  1.54651604, -0.26003285,
        1.89541277]), 114.38860010290168],
       [array([-1.39481912, -2.6483627 , -0.42768623, -3.05700464, -3.04080583,
        2.62577437,  0.21769552, -1.0872623 ,  1.05394345, -1.57137948,
        1.1950884 , -0.96712176, -2.5645344 ,  1.90845498,  1.38427908,
        1.57705242, -2.33757771, -0.84945263,  0.51133036, -0.56222113,
        2.50331093,  2.52422838, -0.0524182 , -3.60118567, -1.71335832,
        0.84361462, -3.15854406, -1.75160862, -0.77667173,
        1.89541277]), 109.684806532334991]
```

1.2. Population Initializations

```
Spherical:
[[array([ 2.25020397, -4.43253316,  1.91162255, -4.00563955, -3.36271321,
       -1.02657363, -0.25797675, -4.07321379,  5.0549201 ,  1.5089233 ,
       -2.97665281, -3.6318213 ,  2.85657439, -3.93993905, -0.27594767,
       -1.30527425, -2.80293234, -1.72602944, -4.11019749, -3.49662874,
        0.80942729, -1.0106108 , -2.94945799, -3.02900083, -1.15353102,
       -3.70218481,  4.22059433,  0.92704273,
       1.2494727 , -2.99512764]), 252.5202136937675],
       [array([ 0.34750203, -0.97950095, -3.51729198,
       -3.72947524, -2.37921109,
        1.33343186,  1.69293732,  0.75596535,  0.21255201, -4.66570175,
        4.06714624,  5.01300804,  0.69413286, -4.06650353, -0.46722299,
        0.80038016,  3.15052323,  3.74398823, -0.90317228,  2.916128  ,
        2.61410849, -1.21919529,  1.55653129, -0.55575674,  2.95320733,
        4.61051829,  1.30688041, -0.83266499,  4.67587009, -1.47476532]),
        220.332467288544251, [array([-2.75632424,
        4.73375877, -1.22202647, -3.49219201, -3.73124909,
        4.22149728,  5.03655688, -3.78302801, -3.70720864, -3.44683917,
       -1.02461979, -0.6711042 , -2.40984212,  3.21166856,  1.05853653,
        3.69215368,  3.85445931, -4.92469007, -0.63310754, -3.58706989,
        3.31377197,  5.06595465,  0.83927469,  2.38990988, -0.31934903,
        0.85778279,  3.69317674,  1.18737045,
        2.4772539 , -4.93584551]), 314.939882064285],
        [array([-1.91445781,  1.65097439,  3.29277551,  4.82133195, -0.06395172,
       -2.61661084, -4.64890717, -2.66746799, -0.12716134, -1.63649781,
       -0.64548127,  1.75348807, -3.28323536,  1.95798052, -4.1493743 ,
       -4.96324782, -1.54448402, -4.43917906,  4.52218806, -1.69541434,
        1.9435143 , -1.52071051,  3.55942599, -3.19129922, -2.67047256,
        1.7157566 ,  0.9804944 ,  2.62545511,  1.32895324,  2.2721228 ]),
        237.925418387441971, [array([-3.85344077,
        4.31487245, -4.16669623,  2.0787691 ,  4.8481825 ,
        4.29793829,  1.26749143, -2.24725349,  5.07861821,  2.92630423,
       -1.74452446,  3.412176  ,  1.01542168, -1.1512638 , -4.87433014,
        0.02679452, -1.000514  ,  1.46722421, -2.87934427, -4.05998772,
        4.98817716, -3.1315208 ,  4.55907543,  3.87113133, -2.02489657,
       -0.9955172 ,  4.71443925, -0.12278362, -0.66790338,  1.00818396]),
       306.12369284133666], [array([-2.23855627,
       2.88023514, -2.66153065, -4.41038409, -1.05066618,
        2.48412749,  1.07343155,  2.52301979, -1.56847759, -3.90266757,
       -0.4935113 ,  4.82374379, -4.0163091 , -5.06627508, -4.56847003,
       -2.47869452,  0.09976609, -0.63383176, -4.13190525,  0.12854109,
```

```
        3.39645745, -2.14747332, -3.21258655, -5.00213374, -2.91646261,
       -2.05349073, -2.30534163,  3.08186754,  0.45024818, -4.62998065]),
      282.81552449741014], [array([-1.24680735,
      -3.4765129 , -2.10044897, -0.45961135, -2.6201763 ,
       0.90088935, -1.50968951,  0.55627113,  2.49357848,  1.65679869,
      -3.39151303, -3.4118862 , -3.04855774, -0.34631603, -0.99683586,
       1.24312458,  0.33155998, -4.31985312,  0.13813528,  0.75995937,
      -1.8820837 , -1.97313941, -0.81212214,  4.62499166,  3.37079056,
       0.75786004, -0.9466112 ,  4.58028929,
       -1.13470368,  1.28037994]),
       158.18016736242632], [array([-1.36733833,
       0.44443009,  3.26623869,  1.42162713, -4.47865624,
      -3.70355696, -4.85951567, -4.32086273,  3.92461447, -3.47766758,
      -0.56327055,  1.92987499,  1.86001214, -4.30059229,  0.87143449,
      -4.88641521,  3.21274839, -1.26929974, -3.0113664 ,  0.38657418,
      -1.42247283,  0.71382133,  3.28659045,  3.3084056 ,  2.23336749,
      -3.35535714, -4.57963564,  2.12654421, -2.90810117,  4.22143275]),
      282.39555388394666], [array([-2.36475462,
      1.62890471, -4.24690903,  2.40878181,  1.73651398,
       2.36522612, -4.47128686, -3.9062515 ,  0.94571093, -4.18968539,
       4.48010144, -2.92252203,  3.66017983, -0.43581863,  2.29764727,
       2.50482667, -1.55656896,  1.26192096, -4.4044233 ,  1.29893993,
       1.49182964,  2.05170813, -2.82085206,  1.83734532,  1.41734678,
      -1.96047423, -2.51678139,  1.95518392, -0.55101787, -4.41068387]),
      226.78197791430472], [array([ 0.93732724,
      3.81996867, -1.09451657, -1.01893486,  0.97571986,
       1.58628875, -2.7198314 , -1.86685081, -0.61416733,  1.5625276 ,
      -0.26251829,  0.26708227,  2.37709701,  2.85330168, -0.36169366,
       4.69415691, -0.01794549,  0.55219467, -4.54545962,  0.18129581,
      -2.67063588,  1.00806928,  2.8476693 , -0.77162786,  3.83254878,
       2.66586228, -4.01337387,  1.92071634,
       -3.19767662, -1.92941843]), 164.36649485307964]]

Rosenbrock:
[[array([ 1.9963636 ,  0.5582033 ,  1.65218891,  0.07282417,  0.93541309,
        0.87874636,  1.17192303,  1.54594015, -1.17778175,  0.55942396,
        0.57702045,  1.64490666,  1.59098863, -1.92425553, -1.99824668,
        0.06574053,  0.9472597 ,  0.01139921,  1.46040079,  1.33255686,
        1.36991472,  1.06400608,  1.46138461, -1.79444961, -1.75401661,
        1.14406015, -0.62709807,  0.25558056, -0.60720134,  0.79061698]),
        16007.942547754279], [array([ 1.48169600e+00,
       -3.04082006e-01,  7.67569939e-01, -5.33255389e-01,
       -1.62899432e+00,  1.08082593e+00,  1.20522562e+00,  1.54291014e+00,
        1.40494574e+00, -9.05916364e-04, -3.00621147e-02, -3.30035392e-01,
        1.91842359e-01, -1.22616863e+00,  1.03318600e+00, -9.90086680e-01,
        9.90464223e-01,  1.49274116e+00,  1.54829027e+00, -9.51884352e-01,
        1.48044985e+00,  1.08467088e-01,  1.75707917e-01,  1.17728615e+00,
       -6.40488964e-01,  1.96665570e+00,  2.63084918e-02, -1.94548439e+00,
       -2.02551880e+00, -1.31585885e+00]),
       13180.972116718738], [array([-0.51924343,
       -0.36257474, -1.85758007, -0.29141549,  0.14817244,
        0.27646502, -1.68869924, -1.5313979 ,  0.46052077, -1.69403353,
       -1.74919892, -1.63481627, -1.60086336,  1.51002526,  1.5786226 ,
        1.9572741 ,  1.35379673,  1.43764077,  0.48502438, -0.7229119 ,
       -1.235278  ,  0.92573782, -1.16555426,  1.54022885,  1.42928722,
        1.90397246, -1.25773346, -1.90308491, -0.04062194,  1.44087887]),
        18201.21842703027], [array([ 1.47287465,
       -1.95818193, -1.05004773, -0.45170015,  0.67993658,
```

```
        1.42021532,  0.33125449, -1.01126923, -0.23330429,  1.57503778,
       -0.65581007, -1.94322917,  1.02061007, -0.31568188, -0.06439933,
       -0.02000401, -0.6534207 , -0.5400191 ,  0.51306118, -0.02083234,
        1.54018647,  0.13171024, -1.42810008, -0.78380667, -0.17630239,
        1.70314455, -0.4374826 ,  1.45532348, -0.23033238, -0.73220946]),
        11920.653815489632], [array([-1.66180375,
        1.53868631, -0.59400601, -1.63639555,  0.82986979,
       -1.54064942,  0.19938095, -0.18238803, -1.68761466,  1.39383316,
        0.2760795 , -1.15231889, -0.79848887, -1.93624937,  1.50618507,
        1.56925758, -1.41613851,  1.39338018,  0.98727759,  1.86410972,
       -1.77216777,  0.25473443, -1.98023491,  1.09263045,  0.45285248,
       -0.42594854, -0.43349494, -0.68672671,  1.12221149, -0.85801033]),
       12638.976946296052], [array([ 0.90766907,
       -0.21075702,  0.35096116,  0.18381801,  1.76040245,
       -1.53861495,  0.94032567,  0.12320879,  1.14789262, -0.01663041,
        0.97336304, -1.17034609, -1.62761974,  0.7495021 , -0.51191323,
       -0.91892539,  1.96595992, -1.41664545,  0.55699985, -0.36722249,
       -1.23817013, -1.57158349,  1.79532916,  1.50471483,  1.50903433,
        1.76171048,  0.33812446,  0.68530988, -1.08294576, -0.11997086]),
        11200.765996000384], [array([ 0.13883474,
        -0.69939995,  0.39002927, -0.76744686, -1.96404434,
        1.51695578,  1.71297748,  0.72058015,  0.06123117,  1.73908429,
        0.01232757,  0.19792799, -1.46163983,  1.13581207,  1.79911121,
       -2.04087176,  0.94381263,  1.18198175, -0.32575292, -1.348563  ,
       -0.92413062,  0.88412175,  0.04762085,  0.08742262,  0.95233321,
        0.66548667,  1.36496243,  0.61828724,  0.77227889, -0.51370193]),
        9115.450636282638], [array([-1.41447732,
        -0.6386405 ,  0.72321781, -0.72716817,  1.10106912,
       -1.32530247, -1.68102211, -0.30907485, -0.87442825, -1.07136791,
       -1.95218305,  0.82159094,  1.2095331 ,  0.20161286,  0.61655952,
       -0.71527766, -0.5387755 , -0.814642  , -1.66021691,  1.46950822,
        1.94415166, -0.74670466, -1.15582434,  1.81931162,  0.3533922 ,
       -0.24338998, -1.97836963,  0.71855717, -0.14088001, -0.02893694]),
       12087.095360640937], [array([-1.12464312,
       0.14843052, -0.86996434,  0.23260493,  1.30976353,
       -1.32999115, -1.19985031,  0.62616419,  1.60181706,  2.04561743,
       -1.5391755 ,  0.73143345, -1.87913837, -0.77520678,  0.81297232,
        1.76348031, -0.9649813 , -1.14953425,  0.02686808, -0.60973641,
        0.54130956,  1.61486285,  0.6913676 ,  0.25019754,  0.20011794,
       -1.84763523,  1.43959564, -1.4253234 , -1.82444195, -1.04215263]),
       16838.190366699586], [array([ 0.87766791,
       1.66286613, -0.74859933, -1.98461911,  0.77681763,
       -0.60267727,  0.32682147, -1.35984482, -1.36137404,  1.30265784,
        1.62273935,  1.08588217,  1.05286479,  0.42658026, -1.65061767,
        1.72918969,  0.80062519, -1.72525001,  1.92898829,  0.85622686,
       -1.14327506,  1.25583673,  1.98537807,  1.4171826 , -0.59917909,
        1.19368353, -1.30844599, -0.03421425,
        1.82666618, -0.89986229]), 12071.231425349288]]

Rastrigin:
[[array([ 1.52299172,  3.27751527, -3.16427011, -2.31747815, -0.89580364,
        1.75343714,  1.77794101, -2.0246538 ,  1.43138668,  2.83686706,
        4.16730564, -4.1281201 , -4.2307505 ,  0.95448586,  2.1800945 ,
        4.38778125, -4.01540746,  2.11151549, -4.19975217,  1.70954747,
       -0.92227457, -3.25138777,  1.27318733, -3.58122494,  0.87183233,
        0.79133668, -2.19595764, -4.92468039, -0.20469229,  2.49264309]),
        474.71268287578596], [array([ 1.54314191,
        -4.81088607,  2.98736194,  0.58040048, -4.31697142,
```

4.99962232, 2.52850865, 0.44797252, 4.99935379, -1.66749969,
4.95842645, 2.04188424, -3.75025584, -1.77464742, -3.96123434,
-3.91489494, 4.65673054, 1.76443698, -4.14495573, -3.51536735,
3.8990871 , 3.19013592, 0.96355694, 1.22013812, -0.7379259 ,
-0.28541556, 3.56048887, 1.9937652 , -3.89377516, 4.88265573]),
574.1536638171797], [array([-4.89740583,
-2.9319274 , 2.98915795, 2.16997151, -2.65877884,
3.61726254, -2.2739397 , 2.01153941, 2.5309783 , 3.12218764,
0.06432426, 1.32440975, -3.71310897, -0.0447622 , -3.63391442,
4.15944314, 4.3999478 , 4.84585693, -2.12649178, 4.24164014,
-0.73705504, -3.91749797, 2.63038489, 1.34805413, 2.07458869,
-0.84888312, 2.50639737, -2.7154307 , -5.02113769, 0.35888303]),
529.6044903039804], [array([3.07869139,
-3.31703139, -0.51366139, 2.97774848, -2.14771896,
4.03400361, 1.23163004, -0.7964257 , -1.02787405, -4.91365655,
-1.65474601, 0.58956749, 0.79430903, 2.40474045, 0.19421523,
1.14435837, 1.9905674 , 3.88574509, 3.92483393, -2.71544386,
1.74210204, -3.69003182, 4.59118045, -2.28412731, -2.38123144,
0.38057412, -4.88871715, 0.0871291 , -4.97610847, 1.79700975]),
474.4563132571599], [array([2.79867956,
1.94456164, -4.98667245, 3.06526647, -3.980093 ,
0.73948228, -3.99238329, -0.34758321, -2.19259156, 2.51133103,
-4.92553656, 0.32112865, -4.10754017, -4.90768747, 3.81104741,
-0.52840726, -0.11160886, -3.92758888, -0.16835002, -2.72046624,
-4.71752359, 1.5870955 , -3.96817064, -0.32242808, -4.32235552,
4.49265128, -4.95757793, 2.67361613, -3.45483504, 3.64486169]),
586.1614220360884], [array([-2.34610722,
4.41938816, 1.54822383, 3.2768952 , 3.88217848,
0.61613785, -0.98262043, -3.39868373, -0.83141687, -1.87382601,
2.02262644, 0.8271883 , 1.35917204, -4.06956408, -0.56224898,
2.8833297 , -0.98871585, -2.71341559, 0.7430345 , 4.13862512,
1.16516718, 5.03825257, 2.06266652, -0.20105868, -4.34445355,
2.15550552, 0.11270001, -2.51014559, 3.98075466, -2.46312566]),
468.8242866134144], [array([1.14344763,
1.49516775, 4.98699032, 4.63497722, 2.52736138,
-2.13656231, -0.08660287, -1.96792246, 0.27242388, -4.82792923,
4.42100923, -0.05750673, 1.18616857, -4.3503184 , -0.41712316,
-2.09534117, 3.98217554, -2.62557962, -4.06632176, 0.16086011,
-3.09385144, 3.67766495, 4.62581363, 1.36297669, -1.42640781,
2.23341646, 2.17984727, 5.04734978, -0.61921468, 2.87796063]),
534.5689163114273], [array([2.23411789,
-5.04810825, -1.36829917, 1.40041713, 3.58235674,
-3.44214651, 3.24222562, 0.60868339, 1.68961331, 0.75173677,
3.93634778, 0.21411774, -5.076969 , -2.37724052, 2.20437178,
4.08174125, 0.77768818, -4.29005249, -4.1684735 , -3.4349082 ,
3.753045 , -1.3376687 , 1.31522178, -3.59396768, 0.86280477,
3.38865593, -2.10829826, 4.42621518, -0.04008906, -1.7052139]),
585.995135192268], [array([0.2649154 ,
0.58072858, -1.50978202, 4.83424688, -2.952535 ,
2.60865384, 1.97930826, -1.27818747, 4.0212756 , 4.74017423,
-4.95988342, 1.5212067 , 0.89748714, 1.3474724 , -4.24932435,
4.06502917, 3.08221398, -4.75172696, -1.30006923, -3.17218695,
0.63164666, -2.12276411, 2.3991997 , -4.13213761, -2.25481285,
-2.06657859, 3.8436284 , 4.17990019, -1.02873992, 2.85549069]),
512.2570804984193], [array([-3.76803517,
-4.94818081, 2.98863327, 2.52160979, 0.24685098,
-3.95672037, -4.52813957, 4.09762521, -3.72773497, -4.97438643,
0.70251021, -0.81475697, 4.33613144, 3.06091244, 2.24029421,

```
        -2.79484282,  1.79568269,  2.71407598, -2.2708213 , -1.25159113,
        -5.03239155,  3.75358777, -0.02046726,  1.74293132,  0.45752247,
        -0.12251685,  3.22759994,  0.81752113,
        5.1082467 , -0.86886247]), 513.5620056704647]]

Schwefel:
[[array([  102.26766228,    53.70515171,   -92.33542226, -196.75441425,
        -278.02964837,    60.57110979,   347.41282997, -135.0938371 ,
         -38.91814138,    37.70537979, -289.51646879, -380.20626676,
        -170.86011584, -264.58176117, -429.71445065,   323.96731425,
         -29.22120739,   312.8423182 ,   160.33912974, -198.70559946,
         -11.0439348 ,   435.98279001,   305.99438385, -377.17159973,
          97.17454592, -269.1105056 , -185.40675962, -325.49363315,
         -81.36713814, -436.45456696]),
         13762.632668740935], [array([   11.33402053,
         -61.36810587, -194.43779059,   101.74671926,
         502.25455759, -267.43214405, -190.94271712,    73.84327819,
         494.28994647,    25.86964265, -391.77547847, -479.29376411,
         263.26085875, -121.07313253,    39.596583  ,   487.21560935,
         276.5785358 ,   255.22688828, -106.55137802,   154.89385775,
        -178.03039517,    60.69604053,   259.24635525, -245.41682375,
        -386.38397255,   -58.83894625,   180.02029796,   428.19665072,
         158.22352775,   335.12433962]), 13970.270245166725], [array([-386.91616134,
102.33043197, -252.49795436, -476.86921642,
         275.44479516,  -23.1721434 , -224.34076723, -239.54063055,
         362.24704582, -181.32036978, -346.55742167,   393.12326066,
        -502.69548208,   485.68978506,   -96.09073645, -413.00872555,
        -281.34744006, -306.37262988, -384.8554747 , -115.70346301,
         162.04135739, -265.19937934, -237.02102542,   503.72276292,
         234.33148768,   195.35809495,   313.71101676, -150.29405214,
         -55.58272513, -279.75140647]), 12699.730182573609], [array([  -70.87557936,
451.91034608, -357.66731898,   463.41498368,
          16.84361795, -359.71193753,   258.19927847, -339.21347268,
        -458.06235774,   210.66339065, -153.88820535,   364.05507694,
        -386.54780214,   462.48596488, -424.46375009, -210.86953563,
        -440.43033027, -153.52749153, -238.26461418,   420.73639087,
        -386.14856548, -493.4247791 ,   213.28063639,   191.06971579,
         -81.24588587,   -47.51689044, -500.35309805,   -19.89928983,
         286.56407568,   151.56276801]),
         12724.53111313521], [array([  308.27991585,
         -415.32420058, -231.68834715, -205.73665364,
        -203.91428035,   468.63352785,    30.30483297, -158.14521783,
         284.55573478,   371.24829397,   499.48027109, -210.2411297 ,
         332.55060762, -209.71849937,   -34.01063794,   495.71917701,
         470.44680641,   329.57821493,   136.38830505,   -15.07057546,
         -16.83806103,   428.44995009,   342.80813889, -213.63842446,
         108.10010734, -214.8919356 , -373.39901124,   469.25292458,
         367.32349957,   117.14980035]),
         14986.302807371163], [array([-259.46784996,
         -291.87598557,   306.62582098,   403.15408466,
         425.46434799,   422.10547409, -293.23137959,   -64.06922743,
         -74.50975753, -149.95297651,   440.1578745 , -142.94080218,
         315.14331772, -338.04960743,   301.96770105, -186.52867218,
         -69.97961854, -279.32952484,   -27.23520259,   252.00023636,
         -38.75925649, -150.94870179,   205.00446408,   493.34558086,
        -383.9667059 , -233.87840878, -407.8393676 , -260.57293877,
        -240.10689476, -475.83401884]),
        11667.740528079887], [array([-409.9837763 ,
```

```
        -190.61257473, -115.2654264 ,  -79.30977347,
        -337.28846867,   10.7259139 , -462.5045271 , -376.06679793,
        -435.12974205, -259.11860206, -250.5176758 , -125.94589575,
         309.18118372,   42.19949556, -107.69785773, -453.56722974,
         -36.24193504,  287.10493204, -473.9473764 , -194.99425065,
         230.99710823, -297.04946026,  322.66835069, -218.20411988,
        -381.67025256,  -19.83250187, -143.53298926, -307.19142814,
        -126.44380062,   98.27232066]),
       14310.488195943952], [array([ 302.53714325,
       198.35093358,  176.5454691 , -150.075331  ,
         84.96817125,  104.2796333 ,  130.86599429,  236.49105014,
       -382.54676809,  132.96153656,   -3.15022009, -184.33606452,
        470.04028872, -333.28828445, -346.46067335,  265.87353693,
        494.78346802, -463.20837956,  -29.36744754, -388.42884547,
       -198.48491663, -193.62224006, -230.38884541, -221.16618144,
       -435.48685393,  507.75897951, -248.39211267, -312.03631084,
       -439.13247887, -409.10636759]),
       15272.078842630048], [array([-462.67867515,
       -104.50209002,  191.9798676 ,  -26.50905492,
       -224.12760685,  341.69447064,  482.4784745 , -404.97629633,
       -221.0553008 , -212.3181101 ,   19.1335406 ,  374.14720176,
         93.51394197,  207.9227905 , -126.61335703, -498.16545522,
       -441.83235237, -365.53309933,  242.85908932,  362.60984883,
        -56.043285  ,  135.69548233, -276.35843191, -137.01376037,
        394.25536622, -305.69992704, -166.4706142 ,  440.8417626 ,
       -108.09878259,    1.36834747]),
       12047.413586872946], [array([ 476.40112173,
       2.0083133 , -400.39146331, -103.22888561,
        246.83559451,  290.3418135 ,  -15.11439123,   25.80536362,
       -354.04479601, -105.63246824,  -33.77201768, -381.86219956,
       -361.62011027,  227.77518301, -368.94642361, -432.28660278,
       -433.72076499,  174.10201355,  302.00131415, -139.39343654,
        230.42823677,  127.18603097,  504.8801558 , -468.94282109,
        505.0539542 ,  415.8013644 ,  115.62385082,   45.65791185,
         77.42614114,  490.07634138]), 14511.540019950846]]

Ackley:
[[array([  18.68955693,   -3.39596404,   23.65662197,  -20.68145344,
         21.39143977,   10.99313732,  -19.47283557,    8.20884835,
        -15.20578526,   20.37391016,   23.09960045,   27.68012977,
        -29.53616365,  -17.32743345,   26.70725514,  -18.22512818,
         -8.87540571,    6.9382119 ,   19.13145633,   12.79845946,
        -27.20001773,   18.92694454,  -10.88878976,  -17.21028499,
         23.66973399,   10.52993765,    7.01544429,   -0.16746119,
         10.70114955,  -10.3646444 ]),
         21.17595472589088], [array([  10.52872845,
         -20.85534017,  -18.02434143,   -6.62690068,
         14.4568607 ,  -20.36411782,   -9.70449912,   -9.3504642 ,
        -24.25715259,  -23.10470866,  -13.72719722,   20.5003105 ,
         16.94310048,   16.22811196,  -16.25381004,  -20.12657574,
         -0.24083136,    6.72435618,   -3.08417826,   13.40650557,
         18.06373081,   20.29283633,   22.35122816,  -19.39994981,
          3.97574291,   -2.85641058,  -13.08158632,   -1.78067841,
         10.19792352,   11.16662    ]),
         20.715007478186], [array([  19.67576422,
         14.67490955,    9.22294105,  -25.27749032,
         13.7227969 ,  -29.47563034,   -4.39283198,  -13.01893122,
        -29.99650705,  -21.39967902,   -2.45818771,   20.44453602,
```

```
 21.88381018, −21.18425793, −11.69836908, −25.59957414,
 18.75123146,   0.33273677,   9.0859173 ,  11.60767577,
 27.66628283, −23.25644239,  13.5429404 ,  18.10126557,
 21.87495687,   0.25095855,  −4.18599174,  26.65795004,
 12.23613561,   5.17680384]),
 21.296836603349554], [array([ −9.41483928,
 −20.47806872, −19.22243189,  20.28563366,
 10.43775244,  26.86431065, −29.90859573,   1.62248869,
−12.81670363,  17.91659935, −12.83182081,  17.08185044,
  1.12694229, −14.78253788, −16.82686255, −15.17541758,
−12.55900769, −28.93150786, −21.60086701, −25.59988554,
  0.8773788 ,  27.40358071,  14.80345893, −19.8413305 ,
  2.6741979 ,  14.10816989,  15.52200783,   7.39442791,
 22.64623465,  −6.80108154]),
 21.126738518023803], [array([ −5.98996939,
 10.99215359, −11.18103682,  13.4177468 ,
 22.88721729,  −1.71706466,  −2.28778873,   0.1026849 ,
 29.56111382,   5.76494803,   2.23350347,  26.21612597,
 10.7710109 ,  10.91412759,  28.31424255,  16.60094673,
−18.77389145,   1.71848423,  −1.67935996, −29.74234356,
−23.37536856,  19.54297452,   4.60229988,   6.74811579,
−20.52771813, −21.19587767,  −2.60717568,   5.22070357,
 20.85752746, −17.14648277]),
 20.955047386988557], [array([−20.99052268,
 6.49774665,  27.75805776,  24.13553847,
−27.1497418 ,  24.66639667,  −7.38975425,   0.93677803,
−24.18190973,   0.73677922, −16.41637448, −16.01018775,
  7.03932992,  26.39766858,   4.13516294, −25.78185397,
−17.82996232,   7.19702213,  22.27798859,  16.29334917,
−13.77768687,   6.69243822,   2.13005941, −15.40456386,
 29.27794095, −25.99866586, −26.10868587,  26.05832962,
−25.59425415,  29.45132463]),
21.225663441928678], [array([−25.66470802,
6.69304443,   2.46003394,  26.66813148,
 −3.65813726,   9.26265765,  21.37484687,  22.94529136,
 25.85036014,  −3.98568408, −10.70745492, −13.61972166,
 21.22257761,  20.74607568,  12.90361837,  25.81837292,
  4.95940883, −15.19516049,   7.43488635, −13.52895573,
−10.0067417 ,  24.45925353,  −7.59837819,  13.2624034 ,
 16.32732787, −20.4832143 ,   1.90157152,  15.41479258,
 22.17159651, −11.81925954]),
 21.072575445742054], [array([ 18.74987568,
 6.09125944,  −5.77162549,   3.48259118,
 −7.86324065,   4.1278906 , −18.36052984,  10.97264937,
−26.71368994, −11.61618564, −17.47526371, −11.8546087 ,
−17.54438619,  25.14378264,  −2.3022616 ,  22.11740103,
  4.04510715,  28.47024585, −20.52522314,  27.17433623,
 23.03867188, −27.98957189,  29.33589938,  −2.41346669,
 −1.94634568,  −8.1303714 , −29.54421516,   5.29615483,
  0.14721885,  21.77413752]),
  21.07438163125648], [array([ 15.86728369,
  −19.46687602,  13.11008063, −10.97150411,
  3.58406331,  26.28400966,  −5.2235405 ,  18.55509263,
 12.87944817,   6.18258296,   1.27415063,  21.32208201,
−29.56121058, −27.80522381,  −2.63415978, −18.83740943,
 29.87123518,   6.60675947,  28.08990633,  −9.09101368,
 −4.68789387, −27.80947212, −26.73655305, −22.33541657,
−11.2074618 ,  27.9948195 ,  11.70989981, −11.81582311,
```

```
        -21.52231014,  26.43247149]),
        21.27396845821991], [array([ 19.62850996,
        -20.86964614,  19.9807346 , -28.68653733,
        -19.05238505, -17.56797271, -28.7072359 ,  25.42818646,
          7.65389804,   1.352517  , -21.30760225,  22.08354003,
         23.17053707, -13.67917902,  -2.82933959, -24.6508506 ,
         24.2511977 ,   1.63100592, -13.56434033, -14.04667534,
         16.24061645, -22.30685717, -25.55113435,  21.06516794,
         29.06755823,  14.88173843,  20.60027988, -29.67247527,
         10.01359533,  22.22710254]), 21.3943366596383 13]]

Griewangk:
[[array([-348.14253908,  395.39232182, -426.04508666, -386.06262   ,
         583.92777114,   60.50005096,  476.1780393 ,  525.16217241,
         155.15821739,  526.81321825, -301.37804425,   17.41649835,
        -532.38559531,  128.39296103,  389.23707944, -176.90944068,
         341.02016773, -378.98618099,  349.10081681,   21.36687456,
         133.10745151,  113.76490375,  544.95225739, -478.97428818,
         -96.45290461,   32.99811858, -152.80762292, -369.66978228,
        -268.13601662, -118.59139732]), 882.5977795544278], [array([ -92.86193707,
89.30823283, -435.17537211,  285.39632255,
         541.98029509, -407.52185269, -285.71910597,   89.8657967 ,
         575.26789728, -379.47842002, -214.01793344,  529.63210364,
         445.97236505, -133.85668146,  189.96172347,  267.01848384,
        -188.15179038, -422.46173628,  492.57141453, -546.14025703,
        -448.96536316, -305.49463153, -559.37775029,    1.84072892,
        -308.2762754 ,  207.9494627 ,   14.78933351,  229.20600468,
         433.84729673,  428.90508704]),
         973.0472537117754], [array([ 360.63398181,
         -414.91165531,  399.53194152,  241.035838,
        -350.64804549,  368.27817289,  529.71958016,  466.68388815,
         -30.84436445, -410.46796143, -271.06565499, -269.44163336,
         320.29975669, -441.04772641,   -5.81401151, -407.22573222,
          -8.95391603,  303.74425067,  -96.62702862,  428.47715707,
        -147.41920001,   41.25511337,  179.41228268, -395.63477028,
        -213.13389323,  143.34405148,  366.0331626 ,  198.13224729,
          36.87011183, -410.76163257]),
          741.5298718581053], [array([-593.32258366,
          590.73195078,   72.98033327,  358.94421496,
         133.66825289, -556.39859801,  305.24093224,  -75.24451588,
          82.66515562,  405.7253636 ,  357.08214843, -500.80304441,
        -487.25235984,  390.09144607,  486.17202982, -177.65390181,
         -85.16519799,  444.00896299, -272.9879982 , -448.19603121,
         580.34395609,  475.18449341,  430.51732089,   96.73505622,
        -384.87033127,  -41.49041106, -227.91305508, -152.68944936,
        -508.70647943,  200.67502738]),
        1058.4927291501667], [array([  -9.84471475,
        -592.47522438,  203.94106865,  238.99849367,
         371.46526802, -407.88284931,  353.81933687, -437.98756246,
        -198.49343532,  546.94790337,  308.25201131,  454.57252182,
         512.35927647,  409.18763377,   19.29444692,  319.91944013,
        -406.32966287,  451.37603677,  171.99922142, -494.12966369,
        -114.14860854, -189.23916561, -372.04426597,  216.24548522,
         383.68337215,   33.08092141,  202.72885486, -104.69192643,
        -350.23091185,  216.47882429]),
        873.6475937831984], [array([ 537.51116178,
        227.62923493, -390.0672834 ,  -51.1800791 ,
         304.01327875,  124.84028934, -296.33624646,  411.47990337,
```

```
−541.6965015 , −120.72985249,  410.74187111, −339.61639783,
−533.17197196,   35.59548844,  290.38897723,  −51.52329144,
−373.96567978,  239.80130076, −318.21898405,   54.98144939,
 144.39584469, −458.33736995,  522.63685999,   42.6716182 ,
−427.48522983,  362.28406066, −212.63646579,  396.41198694,
−396.70363648, −377.24289828]),
863.1987150953094], [array([  44.93180259,
−397.60771005, −455.18406028, −474.89859095,
 141.1062451 ,   25.84982173,  214.39075539,  −71.59120637,
−291.42706707,  403.44797005,  480.00411635,  449.63039337,
 495.14345542,  242.98004875,  465.54257519, −440.02386409,
−264.16835377, −108.74455528,  317.00115685,  453.23007623,
−270.20045845,  325.2843194 ,  567.33200119, −595.83338363,
−194.44424342,  492.4873885 ,  322.45735598,  −13.10965321,
 −78.05521594, −304.38731342]),
 947.4486257932061], [array([  29.69622844,
 −396.3446555 , −147.82813666,  −17.63388533,
 495.58824067, −592.84518191,  449.25064201,  237.55734194,
−592.85313421, −459.96888081, −325.44353651,  260.29419082,
 597.77004188, −198.51531722,  483.30256255,  541.115475  ,
−263.87428505,  448.99251625, −560.81829612, −380.75770354,
  22.63227408,  217.52810187,  149.50504103,  323.89149502,
−449.26271033, −412.04662791,  402.78777915,  563.85854078,
 558.55294282, −539.48981009]),
 1264.9838525739606], [array([ 421.07411562,
 −578.27692703, −133.92026217,  503.27254901,
  49.83228053,  268.96811722,   53.62803963,   −5.15392552,
−540.98777225, −460.4021089 ,  424.32847838, −483.84132411,
 275.49868807,  342.77955776,  447.62100397,  503.56777065,
 −23.41870905, −135.47368925,  146.00905385, −544.72730332,
 123.27673767, −231.19540381,  192.50344364, −523.95369727,
−479.94356176, −436.38488348,  536.74997187, −166.3344066 ,
−353.70713256,  577.77799285]),
1085.86386821579], [array([−378.33978362,
−475.2132017 , −389.92266037, −511.15138956,
−317.70622601,  521.0559668 , −327.49583407, −117.99117157,
 215.85820169, −585.97685032,  500.2454324 ,  591.76267155,
−291.87498302,  143.75000538, −348.4933561 ,  348.46167431,
−489.1821262 ,  410.68257355, −584.99573343,  126.5100543 ,
 243.77346563, −205.05210197, −501.04658089,  524.16535237,
  86.84957787, −595.35796972,  246.14498465,  −75.62078889,
−335.71325916, −216.99628545]), 1150.1400024365614]]
```

2. Code

```
#!/Library/Frameworks/Python.framework/Versions/3.4/bin/python3
# Joel Doumit
# CS472 - Evolutionary Algorithms
# Assignment 2a - Island Model
# Fitness functions used taken from the DEAP Github page,
# https://github.com/DEAP/deap/blob/master/deap/benchmarks/__init__.py

import numpy as np
import csv
import sys
import math
from functools import reduce
from operator import mul
```

```
# sys.argv[1] is the way to get commandline arguments.

populationSize = 10
numberOfGenes = 30

def initializeBoard(x, y, z):
    return np.random.uniform(x, y, z)

def initializePopulation(x, y, z, size):
    return [initializeBoard(x, y, z) for i in range(size)]

def PickMigrants(population):
    chosenIndiv = np.random.choice(populationSize, 2, replace=False)
    selectPool = []
    for i in chosenIndiv:
        selectPool.append(population[i])
        del population[i]
    return selectPool

def selectionPool(population):
    chosenIndiv = np.random.choice(populationSize, 5, replace=False)
    selectPool = []
    for i in chosenIndiv:
        selectPool.append(population[i])
    return selectPool

def sortPool(to_sort):
    to_sort.sort(key=lambda x: x[1])
    return to_sort

def pickParents(sorted_pool):
    return sorted_pool[0][0], sorted_pool[1][0]

def crossover(parents):
    kids = [[],[]]
    crossover = np.random.choice(range(1,numberOfGenes-1))
    for kid in range(2):
        j = 0
        for i in range(numberOfGenes):
            if i < crossover:
                kids[(0+kid)%2].append(parents[(0+kid)%2][i])
            else:
                kids[(0+kid)%2].append(parents[(1+kid)%2][i])
    return np.asarray(kids)

#### SPHERICAL FUNCTIONS ####
def pairSphereIndividuals(population):
    sphereFitnessPop = []
    for i in population:
        sphereFitnessPop.append([i, sphereFitness(i)])
    return sphereFitnessPop

def sphereFitness(individual):
    return sum(gene * gene for gene in individual)

def sphereMutation(kids):
    #swap two values inside the kid for permutation, pick one
```

```
    #value to swap to another random value in combination.
    for i in kids:
        for j in range(numberOfGenes):
            if(np.random.choice([0,1], p=[0.875, 0.125]) == 1):
                i[j] = np.random.uniform(-5.12, 5.12)
    return kids

def sphereGeneration(population):
    newPopulation = []
    parents = pickParents(sortPool(selectionPool(population)))
    kids = (crossover(parents))
    sphereMutation(kids)
    newPopulation.append([kids[0], sphereFitness(kids[0])])
    newPopulation.append([kids[1], sphereFitness(kids[1])])
    return newPopulation

#### ROSENBROCK FUNCTIONS ####
def pairRosenIndividuals(population):
    rosenFitnessPop = []
    for i in population:
        rosenFitnessPop.append([i, rosenFitness(i)])
    return rosenFitnessPop

def rosenFitness(individual):
    return sum(100 * (x * x - y)**2 + (1. - x)**2 \
for x, y in zip(individual[:-1], individual[1:]))

def rosenMutation(kids):
    #swap two values inside the kid for permutation, pick one
    #value to swap to another random value in combination.
    for i in kids:
        for j in range(numberOfGenes):
            if(np.random.choice([0,1], p=[0.875, 0.125]) == 1):
                i[j] = np.random.uniform(-2.048, 2.048)
    return kids

def rosenGeneration(population):
    newPopulation = []
    parents = pickParents(sortPool(selectionPool(population)))
    kids = (crossover(parents))
    rosenMutation(kids)
    newPopulation.append([kids[0], rosenFitness(kids[0])])
    newPopulation.append([kids[1], rosenFitness(kids[1])])
    return newPopulation

#### RASTRIGIN FUNCTIONS ####
def pairRastIndividuals(population):
    rastFitnessPop = []
    for i in population:
        rastFitnessPop.append([i, rastFitness(i)])
    return rastFitnessPop

def rastFitness(individual):
    return 10 * len(individual) + sum(gene * gene - 10 * \
math.cos(2 * math.pi * gene) for gene in individual)

def rastMutation(kids):
    for i in kids:
```

```
        for j in range(numberOfGenes):
            if(np.random.choice([0,1], p=[0.875, 0.125]) == 1):
                i[j] = np.random.uniform(-5.12, 5.12)
    return kids

def rastGeneration(population):
    newPopulation = []
    parents = pickParents(sortPool(selectionPool(population)))
    kids = (crossover(parents))
    rastMutation(kids)
    newPopulation.append([kids[0], rastFitness(kids[0])])
    newPopulation.append([kids[1], rastFitness(kids[1])])
    return newPopulation

### SCHWEFEL FUNCTIONS ###
def pairSchwefelIndividuals(population):
    schwefelFitnessPop = []
    for i in population:
        schwefelFitnessPop.append([i, schwefelFitness(i)])
    return schwefelFitnessPop

def schwefelFitness(individual):
    return (418.9828872724339*30-sum(x*math.sin(math.sqrt(np.abs(x))) for x in individual))

def schwefelMutation(kids):
    for i in kids:
        for j in range(numberOfGenes):
            if(np.random.choice([0,1], p=[0.875, 0.125]) == 1):
                i[j] = np.random.uniform(-30, 30)
    return kids

def schwefelGeneration(population):
    newPopulation = []
    parents = pickParents(sortPool(selectionPool(population)))
    kids = (crossover(parents))
    schwefelMutation(kids)
    newPopulation.append([kids[0], schwefelFitness(kids[0])])
    newPopulation.append([kids[1], schwefelFitness(kids[1])])
    return newPopulation

### ACKLEY FUNCTIONS ###
def pairAckleyIndividuals(population):
    ackleyFitnessPop = []
    for i in population:
        ackleyFitnessPop.append([i, ackleyFitness(i)])
    return ackleyFitnessPop

def ackleyFitness(individual):
    return 20 - 20 * np.exp(-0.2*math.sqrt(1.0/30 * sum(x**2
    for x in individual))) \
+ np.e - np.exp(1.0/30 * sum(math.cos(2*math.pi*x) for x in individual))

def ackleyMutation(kids):
    for i in kids:
        for j in range(numberOfGenes):
            if(np.random.choice([0,1], p=[0.875, 0.125]) == 1):
                i[j] = np.random.uniform(-30, 30)
    return kids
```

```
def ackleyGeneration(population):
    newPopulation = []
    parents = pickParents(sortPool(selectionPool(population)))
    kids = (crossover(parents))
    ackleyMutation(kids)
    newPopulation.append([kids[0], ackleyFitness(kids[0])])
    newPopulation.append([kids[1], ackleyFitness(kids[1])])
    return newPopulation

### GRIEWANGK FUNCTIONS ###
def pairGriewangkIndividuals(population):
    griewangkFitnessPop = []
    for i in population:
        griewangkFitnessPop.append([i, griewangkFitness(i)])
    return griewangkFitnessPop

def griewangkFitness(individual):
    return 1.0/4000.0 * sum(x**2 for x in individual) - \
reduce(mul, (math.cos(x/math.sqrt(i+1.0)) for i, x in enumerate(individual)), 1) + 1

def griewangkMutation(kids):
    for i in kids:
        for j in range(numberOfGenes):
            if(np.random.choice([0,1], p=[0.875, 0.125]) == 1):
                i[j] = np.random.uniform(-600, 600)
    return kids

def griewangkGeneration(population):
    newPopulation = []
    parents = pickParents(sortPool(selectionPool(population)))
    kids = (crossover(parents))
    griewangkMutation(kids)
    newPopulation.append([kids[0], griewangkFitness(kids[0])])
    newPopulation.append([kids[1], griewangkFitness(kids[1])])
    return newPopulation

if __name__ == "__main__":
    method = sys.argv[1]

    if method == "spherical":
        newSpherePop1 = []
        newSpherePop2 = []
        newSpherePop3 = []

        initPop1 = initializePopulation(-5.12, 5.12, numberOfGenes, populationSize)
        initPop2 = initializePopulation(-5.12, 5.12, numberOfGenes, populationSize)
        initPop3 = initializePopulation(-5.12, 5.12, numberOfGenes, populationSize)
        spherePopulation1 = pairSphereIndividuals(initPop1)
        spherePopulation2 = pairSphereIndividuals(initPop2)
        spherePopulation3 = pairSphereIndividuals(initPop3)

        for i in range(50):
            for g in range(5): #the range should be the population size/2
                newSpherePop1.extend(sphereGeneration(spherePopulation1))

            for g in range(5): #the range should be the population size/2
                newSpherePop2.extend(sphereGeneration(spherePopulation2))
```

```
            for g in range(5): #the range should be the population size/2
                newSpherePop3.extend(sphereGeneration(spherePopulation3))

            spherePopulation1 = newSpherePop1
            spherePopulation2 = newSpherePop2
            spherePopulation3 = newSpherePop3
            newSpherePop1 = []
            newSpherePop2 = []
            newSpherePop3 = []

        pop1Migrants = PickMigrants(spherePopulation1)
        pop2Migrants = PickMigrants(spherePopulation2)
        pop3Migrants = PickMigrants(spherePopulation3)

        print ("Migrants from population 1:")
        print(pop1Migrants)

        spherePopulation2.extend(pop1Migrants)
        spherePopulation3.extend(pop2Migrants)
        spherePopulation1.extend(pop3Migrants)

        print("Population 2:")
        print(spherePopulation2)

    elif method == "rosenbrock":
        newRosenPop1 = []
        newRosenPop2 = []
        newRosenPop3 = []

        initPop1 = initializePopulation(-2.048, 2.048, numberOfGenes, populationSize)
        initPop2 = initializePopulation(-2.048, 2.048, numberOfGenes, populationSize)
        initPop3 = initializePopulation(-2.048, 2.048, numberOfGenes, populationSize)
        rosenPopulation1 = pairRosenIndividuals(initPop1)
        rosenPopulation2 = pairRosenIndividuals(initPop2)
        rosenPopulation3 = pairRosenIndividuals(initPop3)

        for i in range(50):
            for g in range(5): #the range should be the population size/2
                newRosenPop1.extend(rosenGeneration(rosenPopulation1))

            for g in range(5): #the range should be the population size/2
                newRosenPop2.extend(rosenGeneration(rosenPopulation2))

            for g in range(5): #the range should be the population size/2
                newRosenPop3.extend(rosenGeneration(rosenPopulation3))

            rosenPopulation1 = newRosenPop1
            rosenPopulation2 = newRosenPop2
            rosenPopulation3 = newRosenPop3
            newRosenPop1 = []
            newRosenPop2 = []
            newRosenPop3 = []

        pop1Migrants = PickMigrants(rosenPopulation1)
        pop2Migrants = PickMigrants(rosenPopulation2)
        pop3Migrants = PickMigrants(rosenPopulation3)
```

```
    print ("Migrants from population 1:")
    print(pop1Migrants)

    rosenPopulation2.extend(pop1Migrants)
    rosenPopulation3.extend(pop2Migrants)
    rosenPopulation1.extend(pop3Migrants)

elif method == "rastrigin":
    newRastPop1 = []
    newRastPop2 = []
    newRastPop3 = []

    initPop1 = initializePopulation(-5.12, 5.12, numberOfGenes, populationSize)
    initPop2 = initializePopulation(-5.12, 5.12, numberOfGenes, populationSize)
    initPop3 = initializePopulation(-5.12, 5.12, numberOfGenes, populationSize)
    rastPopulation1 = pairRastIndividuals(initPop1)
    rastPopulation2 = pairRastIndividuals(initPop2)
    rastPopulation3 = pairRastIndividuals(initPop3)

    for i in range(50):
        for g in range(5): #the range should be the population size/2
            newRastPop1.extend(rastGeneration(rastPopulation1))

        for g in range(5): #the range should be the population size/2
            newRastPop2.extend(rastGeneration(rastPopulation2))

        for g in range(5): #the range should be the population size/2
            newRastPop3.extend(rastGeneration(rastPopulation3))

        rastPopulation1 = newRastPop1
        rastPopulation2 = newRastPop2
        rastPopulation3 = newRastPop3
        newRastPop1 = []
        newRastPop2 = []
        newRastPop3 = []

    pop1Migrants = PickMigrants(rastPopulation1)
    pop2Migrants = PickMigrants(rastPopulation2)
    pop3Migrants = PickMigrants(rastPopulation3)

    print ("Migrants from population 1:")
    print(pop1Migrants)

    rastPopulation2.extend(pop1Migrants)
    rastPopulation3.extend(pop2Migrants)
    rastPopulation1.extend(pop3Migrants)

elif method == "schwefel":
    newSchwefelPop1 = []
    newSchwefelPop2 = []
    newSchwefelPop3 = []

    initPop1 = initializePopulation(-512.03, 511.97,
    numberOfGenes, populationSize)
    initPop2 = initializePopulation(-512.03, 511.97,
    numberOfGenes, populationSize)
    initPop3 = initializePopulation(-512.03, 511.97,
    numberOfGenes, populationSize)
```

```
    schwefelPopulation1 = pairSchwefelIndividuals(initPop1)
    schwefelPopulation2 = pairSchwefelIndividuals(initPop2)
    schwefelPopulation3 = pairSchwefelIndividuals(initPop3)

    for i in range(50):
        for g in range(5): #the range should be the population size/2
            newSchwefelPop1.extend(schwefelGeneration(schwefelPopulation1))

        for g in range(5): #the range should be the population size/2
            newSchwefelPop2.extend(schwefelGeneration(schwefelPopulation2))

        for g in range(5): #the range should be the population size/2
            newSchwefelPop3.extend(schwefelGeneration(schwefelPopulation3))

        schwefelPopulation1 = newSchwefelPop1
        schwefelPopulation2 = newSchwefelPop2
        schwefelPopulation3 = newSchwefelPop3
        newSchwefelPop1 = []
        newSchwefelPop2 = []
        newSchwefelPop3 = []

    pop1Migrants = PickMigrants(schwefelPopulation1)
    pop2Migrants = PickMigrants(schwefelPopulation2)
    pop3Migrants = PickMigrants(schwefelPopulation3)

    schwefelPopulation2.extend(pop1Migrants)
    schwefelPopulation3.extend(pop2Migrants)
    schwefelPopulation1.extend(pop3Migrants)

elif method == "ackley":
    newAckleyPop1 = []
    newAckleyPop2 = []
    newAckleyPop3 = []

    initPop1 = initializePopulation(-30, 30, numberOfGenes, populationSize)
    initPop2 = initializePopulation(-30, 30, numberOfGenes, populationSize)
    initPop3 = initializePopulation(-30, 30, numberOfGenes, populationSize)
    ackleyPopulation1 = pairAckleyIndividuals(initPop1)
    ackleyPopulation2 = pairAckleyIndividuals(initPop2)
    ackleyPopulation3 = pairAckleyIndividuals(initPop3)

    for i in range(50):
        for g in range(5): #the range should be the population size/2
            newAckleyPop1.extend(ackleyGeneration(ackleyPopulation1))

        for g in range(5): #the range should be the population size/2
            newAckleyPop2.extend(ackleyGeneration(ackleyPopulation2))

        for g in range(5): #the range should be the population size/2
            newAckleyPop3.extend(ackleyGeneration(ackleyPopulation3))

        ackleyPopulation1 = newAckleyPop1
        ackleyPopulation2 = newAckleyPop2
        ackleyPopulation3 = newAckleyPop3
        newAckleyPop1 = []
        newAckleyPop2 = []
        newAckleyPop3 = []
```