some possible tests I can do:

- a video of the visualisation
- test whether n nodes are generated
- test deleting, adding nodes
- test clear function
- test whether nodes can't be created outside the container
- start button
- alerts when some of the inputs are not valid
- speed adjusting
- fast forward
- stop and start again
- stop and clear paths
- held-karp showing up for smaller instances
- hide checks

Testing

UI (visualisation)

Algorithms

Held-Karp

To test held-karp algorithm, I created a simple test.py file and found some small instances of tsp wih the answers to them here: algorithm - Data for simple TSP - Stack Overflow

```
import common
import held_karp

n = int(input())
dist = []
for i in range(n):
    row = [int(x) for x in input().split()]
    dist.append(row)
tsp_input = common.TSP_input(n, dist, None)
hk = held_karp.held_karp(tsp_input)
print(hk.cost)
print(hk.path)
```

I found 3 tests and performed all of them on my held-karp function

Tests

test 1

11 0 29 20 21 16 31 100 12 4 31 18 29 0 15 29 28 40 72 21 29 41 12 20 15 0 15 14 25 81 9 23 27 13 21 29 15 0 4 12 92 12 25 13 25 16 28 14 4 0 16 94 9 20 16 22 31 40 25 12 16 0 95 24 36 3 37 100 72 81 92 94 95 0 90 101 99 84 12 21 9 12 9 24 90 0 15 25 13 4 29 23 25 20 36 101 15 0 35 18 31 41 27 13 16 3 99 25 35 0 38 18 12 13 25 22 37 84 13 18 38 0

expected: 253

answer:253

test passed

test 2

6 9999 64 378 519 434 200 64 9999 318 455 375 164 378 318 9999 170 265 344 519 455 170 9999 223 428 434 375 265 223 9999 273 200 164 344 428 273 9999

expected: 1248

answer: 1248

test passed

test 3

15 -1 141 134 152 173 289 326 329 285 401 388 366 343 305 276 141 -1 152 150 153 312 354 313 249 324 300 272 247 201 176 134 152 -1 24 48 168 210 197 153 280 272 257 237 210 181 152 150 24 -1 24 163 206 182 133 257 248 233 214 187 158 173 153 48 24 -1 160 203 167 114 234 225 210 190 165 137 289 312 168 163 160 -1 43 90 124 250 264 270 264 267 249 326 354 210 206 203 43 -1 108 157 271 290 299 295 303 287 329 313 197 182 167 90 108 -1 70 164 183 195 194 210 201 285 249 153 133 114 124 157 70 -1 141 147 148 140 147 134 401 324 280 257 234 250 271 164 141 -1 36 67 88 134 150 388 300 272 248 225 264 290 183 147 36 -1 33 57 104 124 366 272 257 233 210 270 299 195 148 67 33 -1 26 73 96 343 247 237 214 190 264 295 194 140 88 57 26 -1 48 71 305 201 210 187 165 267 303 210 147 134 104 73 48 -1 30 276 176 181 158 137 249 287 201 134 150 124 96 71 30 -1

expected: 1194

answer: 1194

test passed

Results

3/3 tests passed

Ant Colony Optimisation

We can't directly test ACO like Held-Karp algorithm with particular inputs and solutions to them, because ACO is a meta heuristic and its output will be different every time you run it as it uses randomisation. Furthermore, ACO's performance depends on how well input parameters are suited to the input, so we can't guarantee its accuracy. But we can:

- compare its output to the well-known TSP instances and their solutions
- visually evaluate its performance

Note that for each test I will write parameters used in the form (alpha, beta, Q, evaporation rate, number of ants, number of iterations)

Since the output may be different every time you run a program, I will it 5 times for each test and take an average for the fairness of this test

Tests

test 1

15 -1 141 134 152 173 289 326 329 285 401 388 366 343 305 276 141 -1 152 150 153 312 354 313 249 324 300 272 247 201 176 134 152 -1 24 48 168 210 197 153 280 272 257 237 210 181 152 150 24 -1 24 163 206 182 133 257 248 233 214 187 158 173 153 48 24 -1 160 203 167 114 234 225 210 190 165 137 289 312 168 163 160 -1 43 90 124 250 264 270 264 267 249 326 354 210 206 203 43 -1 108 157 271 290 299 295 303 287 329 313 197 182 167 90 108 -1 70 164 183 195 194 210 201 285 249 153 133 114 124 157 70 -1 141 147 148 140 147 134 401 324 280 257 234 250 271 164 141 -1 36 67 88 134 150 388 300 272 248 225 264 290 183 147 36 -1 33 57 104 124 366 272 257 233 210 270 299 195 148 67 33 -1 26 73 96 343 247 237 214 190 264 295 194 140 88 57 26 -1 48 71 305 201 210 187 165 267 303 210 147 134 104 73 48 -1 30 276 176 181 158 137 249 287 201 134 150 124 96 71 30 -1

expected: 1194

parameters: (2, 3, 100, 0.6, 25, 20)

answers: 1194, 1219, 1194, 1219, 1219

average: 1209 (1.3% difference)

test 2

ref: TSP - Data for the Traveling Salesperson Problem (fsu.edu) (GR17)

17 0 633 257 91 412 150 80 134 259 505 353 324 70 211 268 246 121 633 0 390 661 227 488 572 530 555 289 282 638 567 466 420 745 518 257 390 0 228 169 112 196 154 372 262 110 437 191 74 53 472 142 91 661 228 0 383 120 77 105 175 476 324 240 27 182 239 237 84 412 227 169 383 0 267 351 309 338 196 61 421 346 243 199 528 297 150 488 112 120 267 0 63 34 264 360 208 329 83 105 123 364 35 80 572 196 77 351 63 0 29 232 444 292 297 47 150 207 332 29 134 530 154 105 309 34 29 0 249 402 250 314 68 108 165 349 36 259 555 372 175 338 264 232 249 0 495 352 95 189

326 383 202 236 505 289 262 476 196 360 444 402 495 0 154 578 439 336 240 685
390 353 282 110 324 61 208 292 250 352 154 0 435 287 184 140 542 238 324 638
437 240 421 329 297 314 95 578 435 0 254 391 448 157 301 70 567 191 27 346 83
47 68 189 439 287 254 0 145 202 289 55 211 466 74 182 243 105 150 108 326 336
184 391 145 0 57 426 96 268 420 53 239 199 123 207 165 383 240 140 448 202 57 0
483 153 246 745 472 237 528 364 332 349 202 685 542 157 289 426 483 0 336 121
518 142 84 297 35 29 36 236 390 238 301 55 96 153 336 0

expected: 2085