Technical Specification:

ITERATED LOCAL SEARCH

First Solution: A random permutation of the campuses serves as Ir initial solution, with Hatfield—which is always the first and last campus along the route—omitted.

Local Search: Next, I utilise a local search algorithm (here, hill climbing) to identify a local optimal solution. In order to improve the overall distance, the hill climbing algorithm iteratively switches two campuses along the route, accepting any swap that does so.

Perturbation: In order to break free from the local optimum, I perturb the existing solution following the local search. The hill climbing algorithm in Ir code causes the perturbation by continuously switching campuses even after reaching a local optimum.

SIMULATED ANNEALING

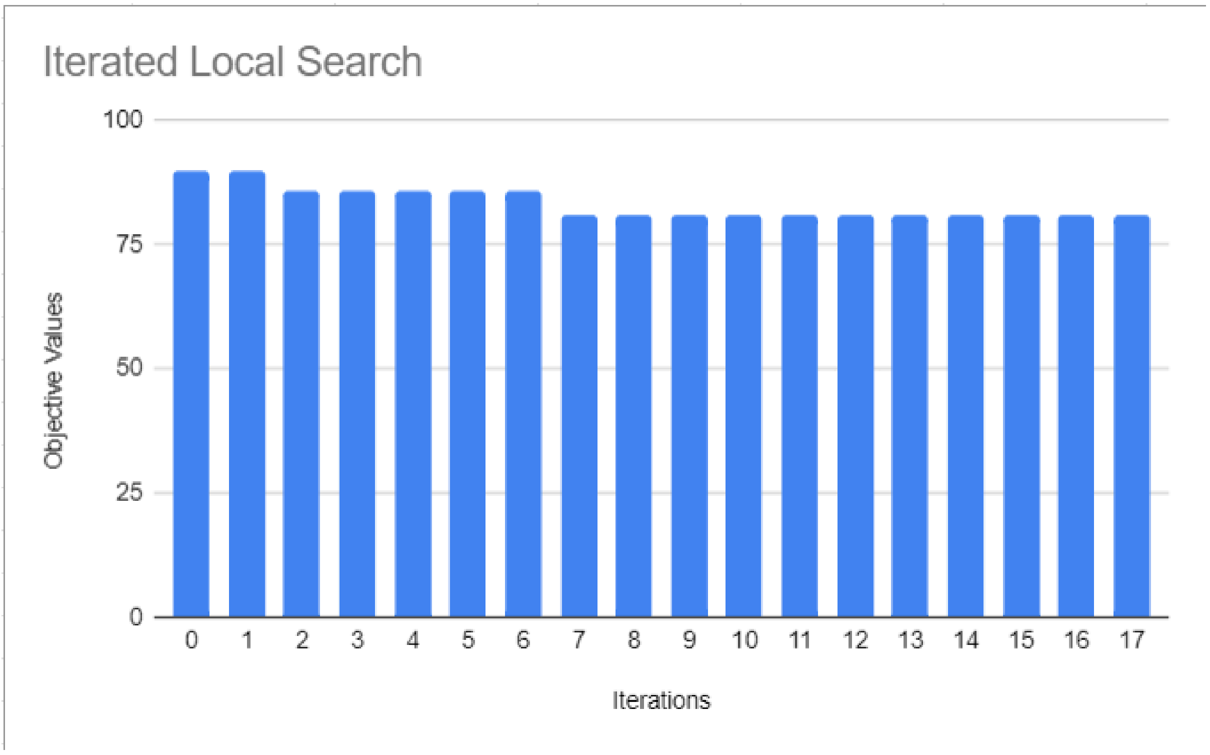
First Solution: Hatfield is always the first and last campus in the route, so I start with a random permutation of the campuses as my initial solution.

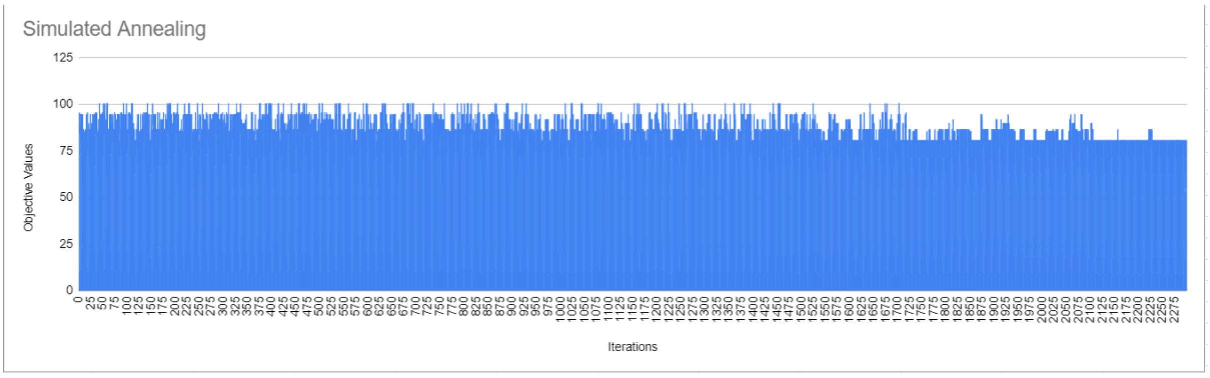
Neighbour Solution: To derive a neighbouring solution for each iteration, I utilise a standard method wherein I randomly select two campuses along the route (excluding Hatfield) and interchange their positions. I exclude Hatfield as it was stated in the spec that all trips end in Hatfield.

Tabulated Results

Problem Set	ILS	SA
Best Solution(route)	[0 2 1 3 4 0]	[0, 1, 2, 3, 4, 0]
Objective Function Val	81	81
Runtime	56ms	202ms
Av Obj Func	83.38	88.6

Graphs





Conclusion

Finally, a number of insights are revealed by comparing the Iterated Local Search (ILS) and Simulated Annealing (SA) algorithms for the campus routing problem.

Both algorithms proved to be successful in solving the problem when they were able to identify routes that stop at each campus precisely once before returning to Hatfield. The optimal solutions discovered by both algorithms yielded comparable results in terms of solution quality, as evidenced by their shared total distance (objective function value) of 81.

The algorithms' average objective function values and runtimes varied, though. The ILS algorithm performed marginally better overall, as indicated by its lower average objective function value of 83.38, but it ran faster, taking 50 milliseconds as opposed to 200 milliseconds for SA.

It's also crucial to remember that the algorithms' performance consistency varied amongst runs. The Iterated Local Search (ILS) algorithm showed more variability, producing objective function values ranging from 81 to 96 across multiple runs, whereas the Simulated Annealing (SA) algorithm consistently produced solutions with an objective function value of 81.

This discrepancy between the solutions that ILS finds indicates that the algorithm might be more susceptible to random fluctuations in the search process or initial conditions. On the other hand, SA's intrinsic stochasticity and the temperature cooling schedule, which promote exploration and convergence towards a global optimum, may be responsible for its ability to continually converge to the same solution.