**Algorytm Genetyczny**

Naszym tematem projektu jest implementacja algorytmu genetycznego, który jest używany do znajdowania optymalnej trasy pomiędzy zestawem miast. Algorytm ten jest często stosowany w problemach optymalizacyjnych, takich jak problem komiwojażera. Kod wykorzystuje koncepcje algorytmów genetycznych, takie jak selekcja, krzyżowanie, mutacja i ewolucja populacji, aby efektywnie znaleźć rozwiązanie problemu optymalizacyjnego. Algorytm ten jest szczególnie użyteczny w problemach, gdzie przestrzeń rozwiązań jest zbyt duża

**Mapa wybranych punktów do określenia najlepszej trasy**



**Custom\_city\_list** tworzy słownik o nazwie gdzie każdemu miastu przypisane są współrzędne (x, y).



**create\_route** tworzy losową trasę, zaczynając od **MiastoA** i losowo permutując pozostałe miasta.



**Calculate\_distance** oblicza całkowitą odległość trasy, sumując odległości między kolejnymi miastami na trasie oraz odległość powrotną do miasta początkowego.



**rank\_routes** ocenia każdą trasę w populacji według jej długości (im krótsza, tym lepsza).



Funkcje algorytmu genetycznego:

**Selection** Wybiera najlepsze trasy (elity) oraz losowo inne trasy, aby utworzyć nową populację.



**mating\_pool** tworzy pulę do krzyżowania na podstawie wyników selekcji.



**Breed** krzyżuje dwie trasy (rodziców) w celu utworzenia nowej trasy (dziecka).



**Mutate** modyfikuje trasę przez losowe zamienianie miejscami miast, co ma na celu wprowadzenie różnorodności genetycznej.



**next\_generation** tworzy nową generację tras poprzez zastosowanie selekcji, krzyżowania i mutacji.



**Główna funkcja algorytmu**

**genetic\_algorithm** uruchamia algorytm genetyczny, tworząc początkową populację tras, a następnie iteracyjnie poprawiając ją przez określoną liczbę generacji.



Oto ostateczny wydruk programu:

