**Diseño del Individuo**

El individuo generado en nuestra solución, consiste en una clase que contiene las 5 casas del problema. Cada una de esas casas tiene los atributos principales que se necesitan para cumplir con dichas condiciones:

* Color
* Profesión
* Editor utilizado
* Base de datos Utilizada
* Lenguaje Utilizado

**Generación de la población**

El tamaño de la población esta dado por una constante ubicada en la clase Main. Se hicieron pruebas con hasta 200000 individuos.

Cada Individuo se genera, modificando los atributos de las casas aleatoriamente, con los posibles valores para cada uno, verificando a su vez, que los valores entre una casa y otra del mismo individuo, no se repitan.

Luego de generada la población, se procede a verificar el Score (O adaptación) de cada individuo

**Operador Selección**

El calculo de Score de cada individuo, esta dado por la cantidad de condiciones que cumplan con respecto al problema. Es decir, que por cada uno de las 14 pistas que el individuo cumpla, su score aumentara en 1. Es menester aclarar, que por como esta diseñada la solución, la primera pista que indica que tienen que ser 5 casas, siempre se va a cumplir, por lo que no es considerada para el calculo del score.

Luego de calculado el score de cada individuo, se procede a seleccionar los dos individuos con mayor Score. Este paso se realiza con el método seleccionPadres() de la clase Algoritmo.

**Operador Cruzamiento**

Con los padres ya seleccionados, se vuelve a generar una nueva población, utilizando esos dos padres.

Cada individuo que se genera, asigna un valor a los atributos de sus casas de manera aleatoria, considerando los mismos atributos que tienen sus padres.

Esto se logra con el método CrossOver() de la clase Algoritmo, que a su vez consume el método heredar() de la clase Individuo

**Operador Mutación**

Se propone un método mutar() en la clase Individuo, que consiste en buscar los atributos repetidos entre las casas de un mismo individuo, y asignarles un valor que no se repita. Esto provoca que el índice de aleatoriedad crezca mucho.

Con esta implementación, comprobamos que si bien, nunca se repiten los datos en el resultado final y el Score es alto, sin este método, el score suele ser mas alto y mucho más rápido.

**Condiciones de corte**

Se proponen 2 condiciones de corte.

* Con una cantidad de iteraciones máxima, seteada desde el Main
* Cuando algún individuo llegue a tener un score de 14