Escuela Superior de Cómputo

Algoritmo genético **Dra. Miriam Pescador Rojas**

NOMBRE: __García Rodríguez Erick Daniel_ Rodriguez Ramírez Fernanda____

Ejercicio 1. Determinar la aptitud de una población con codificación binaria.

```
def aptitud(individuo):
  aux = 0
  for i in range(len(individuo)):
    aux += individuo[i]*(i**2)
  return aux
```

a) A partir del código 1, indique cuál es el valor de aptitud a cada uno de los individuos, de acuerdo con el siguiente código.

No Individuo	Codificación (cromosoma)	Aptitud
1	1 0 0 1 1 1 0 0 1 0 1 1 0 0 1	531
2	1 1 1 1 1 1 1 1 0 0 0 0 0 0 0	140
3	0 0 0 0 0 0 0 0 1 1 1 1 1 1 1	875
4	1 0 1 0 1 0 1 0 1 0 1 0 1 0 1	560
5	0 1 0 1 0 1 0 1 0 1 0 1 0 1 0	455
6	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	1015

b) Escriba la ecuación matemática que determina el valor de aptitud

$$f(x) = \sum_{i=1}^{n} x_i x_i^2$$

c) Diga de que tamaño es el espacio de búsqueda de este problema y cuál es la solución óptima global

El espacio de búsqueda es de $2^{15} = \mathbf{32768}$
La solución optima global es la del individuo 6, es decir, una cadena de 15 1's.

Ejercicio 2. Operador de selección por ruleta

a) Implementar el siguiente código y simule que gira 4 veces la ruleta, muestre capturas de la ejecución y diga que individuos fueron seleccionados.

```python
import random
def ruleta(poblacion, aptitud):
  # Calcular la aptitud total de la población
  aptitudes = [aptitud(individuo) for individuo in poblacion]
  aptitud_total = sum(aptitudes)
  # Calcular la probabilidad de selección para cada individuo
  probabilidades = [aptitud / aptitud_total for aptitud in aptitudes]
  print('individuos', poblacion)
  print('aptitudes: ', aptitudes)
  print('probabilidades: ', probabilidades)
  # Generar un número aleatorio entre 0 y 1
  aleatorio = random.random()
  print('numero aleatorio entre 0 y 1: ',  aleatorio)
  # Seleccionar un individuo utilizando la ruleta
  acumulado = 0
  for i, probabilidad in enumerate(probabilidades):
    acumulado += probabilidad
    if aleatorio <= acumulado:
      return poblacion[i]
```

Primera tirada: ganó el individuo 3

```
Individuos:
        [1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1]
        [1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0]
        [0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1]
        [1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1]
        [0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0]
        [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]

Aptitudes:  [531, 140, 875, 560, 455, 1015]

Probabilidades:
        0.14848993288590603
        0.039149888143176735
        0.2446868008948546
        0.15659955257270694
        0.1272371364653244
        0.28383668903803133

Numero aleatorio entre 0 y 1 =  0.4127531530386066
[0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1]
```

Segunda tirada: Ganó el individuo 4

```
Individuos:
        [1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1]
        [1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0]
        [0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1]
        [1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1]
        [0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0]
        [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]

Aptitudes:  [531, 140, 875, 560, 455, 1015]

Probabilidades:
        0.14848993288590603
        0.039149888143176735
        0.2446868008948546
        0.15659955257270694
        0.1272371364653244
        0.28383668903803133

Numero aleatorio entre 0 y 1 =  0.4323923814866434
[1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1]
```

Tercera tirada: Ganó el individuo 3

```
Individuos:
        [1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1]
        [1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0]
        [0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1]
        [1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1]
        [0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0]
        [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]

Aptitudes:  [531, 140, 875, 560, 455, 1015]

Probabilidades:
        0.14848993288590603
        0.039149888143176735
        0.2446868008948546
        0.15659955257270694
        0.1272371364653244
        0.28383668903803133

Numero aleatorio entre 0 y 1 =  0.23470569030484245
[0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1]
```

Cuarta tirada: Ganó el individuo 6

```
Individuos:
        [1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1]
        [1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0]
        [0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1]
        [1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1]
        [0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0]
        [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]

Aptitudes:  [531, 140, 875, 560, 455, 1015]

Probabilidades:
        0.14848993288590603
        0.039149888143176735
        0.2446868008948546
        0.15659955257270694
        0.1272371364653244
        0.28383668903803133

Numero aleatorio entre 0 y 1 =  0.9545582000223531
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
```

b) Implemente la versión 2 de la ruleta “con reemplazo”, es decir elimine el individuo seleccionado y muestre su ejecución.

```
Individuos:
        [1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1]
        [1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0]
        [0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1]
        [1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1]
        [0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0]
        [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]

Aptitudes:  [531, 140, 875, 560, 455, 1015]

Probabilidades:
        0.14848993288590603
        0.039149888143176735
        0.2446868008948546
        0.15659955257270694
        0.1272371364653244
        0.2838366890803133

Numero aleatorio entre 0 y 1 =  0.7467460645769717

Individuos:
        [1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1]
        [1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0]
        [0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1]
        [1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1]
        [0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0]

Aptitudes:  [531, 140, 875, 560, 455]

Probabilidades:
        0.207340882467786
        0.05466614603670441
        0.3416634127294026
        0.21866458414681764
        0.17766497461928935

Numero aleatorio entre 0 y 1 =  0.16829840812893815

Individuos:
        [1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0]
        [0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1]
        [1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1]
        [0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0]

Aptitudes:  [140, 875, 560, 455]

Probabilidades:
        0.06896551724137931
        0.43103448275862066
        0.27586206896551724
        0.22413793103448276

Numero aleatorio entre 0 y 1 =  0.2695201367473664

Individuos:
        [1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0]
        [1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1]
        [0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0]

Aptitudes:  [140, 560, 455]

Probabilidades:
        0.12121212121212122
        0.48484848484848486
        0.3939393939393939

Numero aleatorio entre 0 y 1 =  0.26761991434755983

Individuos:
        [1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0]
        [0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0]

Aptitudes:  [140, 455]

Probabilidades:
        0.23529411764705882
        0.7647058823529411

Numero aleatorio entre 0 y 1 =  0.7308678343825012
[0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0]
```

c) Explique en cuál de las dos versiones hay mayor presión de selección.

Hay mayor presión de selección en la ruleta sin remplazo, ya que, al no eliminar al ganador, la aptitud media de la población es mayor que en la ruleta con remplazo, donde la aptitud media disminuye mientras van eliminando a los ganadores.

Ejercicio 3. Operadores de recombinación

a) Implemente el siguiente código y ejecute 5 veces (aleatoriamente) para los individuos 2 y 3 del ejercicio 1, de tal manera que estos sean los padres 1 y 2 en el código.

2	**1111111110000000**
3	**0000000001111111**

```
import random
def cruce_uniforme(padre1, padre2):
  hijo1 = []
  hijo2 = []
  for i in range(len(padre1)):
    if random.random() < 0.5:
      hijo1.append(padre1[i])
      hijo2.append(padre2[i])
    else:
      hijo1.append(padre2[i])
      hijo2.append(padre1[i])
  return [hijo1, hijo2]

def cruce_punto(padre1, padre2):
  hijo1 = []
  hijo2 = []
  punto_cruce = random.randint(1, len(padre1) - 2)

  for i in range(len(padre1)):
    if i < punto_cruce:
      hijo1.append(padre1[i])
      hijo2.append(padre2[i])
    else:
      hijo1.append(padre2[i])
      hijo2.append(padre1[i])

  return [hijo1, hijo2]
```

b) Muestre dos tablas con los resultados de ejecución para cada técnica

CRUZA UNIFORME

No. Ejecución	Hijo 1	Aptitud H1	Hijo 2	Aptitud H2
1	101110000110000	210	010001111001111	805
2	101111111111110	818	010000000000001	197
3	000101110110101	640	111010001001010	375
4	0101001011111001	608	101011010000110	407
5	101010001011001	501	010101110100110	514

CRUZA DE 1 PUNTO

No. Ejecución	Hijo 1	Aptitud H1	Hijo 2	Aptitud H2
1	110000001111111	876	001111110000000	139
2	111111110000111	649	000000001111000	366
3	111111101111111	966	000000010000000	204
4	111111110001111	770	000000001110000	245
5	111111110111111	951	000000001000000	64

c) Conteste a cuál de las cruzas le fue mejor en la aptitud de sus descendientes, explique ¿por qué?

La mejor aptitud se obtuvo con la cruza de 1 punto, ya que el resultado fue el siguiente cromosoma: **111111101111111**. En este caso, el cruzamiento tomó la primera mitad del cromosoma del padre 1 y la segunda mitad del cromosoma del padre 2. Dado que ambas mitades seleccionadas contenían varios genes con valor **1**, la combinación resultante generó un descendiente con alta aptitud.

Ejercicio 4. Operador de mutación.

```
import random
def mutacion(individuo, porcMuta):
  individuo_mutado = []
  for i in range(len(individuo)):
    if random.random() < porcMuta:
      if individuo[i] == 1:
         individuo_mutado[i] = 0
      else:
         individuo_mutado[i] = 1

  return individuo_mutado
```

a) Ejecute 3 veces el código de mutación con diferentes valores del parámetro porcMuta = {0.01, 0.1, 0.2, 0.5} y reporte cómo se modifica el cromosoma del individuo 6.

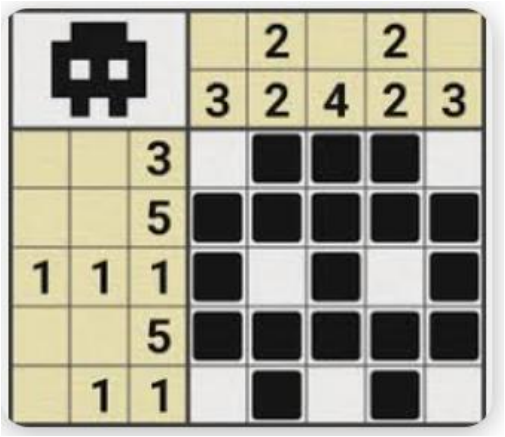
6	**1 1 1 1 1 1 1 1 1 1 1 1 1 1 1**

Ejecución	porcMuta = 0.01	PorcMuta = 0.1	PorcMuta = 0.2	PorcMuta = 0.5
1	111111111111111	110011111111011	100111110111111	111101101111100
2	111111111111111	111111111101111	101111011111111	001101111101110
3	111111111111111	011110111011100	101101000100111	100111100001001

b) Analice los resultados y diga de qué manera afecta el porcentaje de mutación en el cromosoma del individuo, ¿qué valor considera que es mejor usar y por qué?

El efecto del porcentaje de mutación (PorcMuta) depende de su valor. Si es alto, se generan más cambios en el cromosoma, lo que ayuda a explorar más soluciones y salir de malas combinaciones, pero puede romper buenas soluciones encontradas. Si es muy bajo, hay menos cambios, lo que mantiene las buenas combinaciones, pero puede hacer que el algoritmo se quede atascado en una solución que no es la mejor por lo que un valor intermedio suele ser mejor, para equilibrar la exploración y la mejora de las soluciones.

Ejercicio 5. Suponga el siguiente nonograma para resolver con un algoritmo genético, cuyas restricciones (o reglas) podemos definirlas en las listas r y c:
Pr = [(0,0,3), (0,0,5), (1,1,1), (0,0,5), (0,1,1)]
Pc = [(0,3), (2,2), (0,4), (2,2), (0,3)]



Suponga que el problema se resolverá con un algoritmo genético con codificación binaria.

a) Proponga una función objetivo para este problema

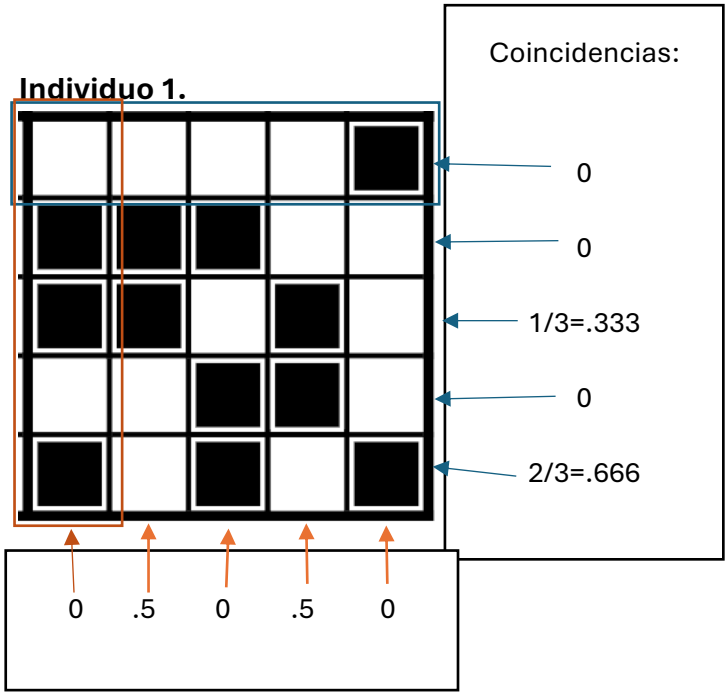
$$g(x) = \sum_{r=0}^{n} x_r + \sum_{c=0}^{n} x_c$$

x_r coincidencias en las filas

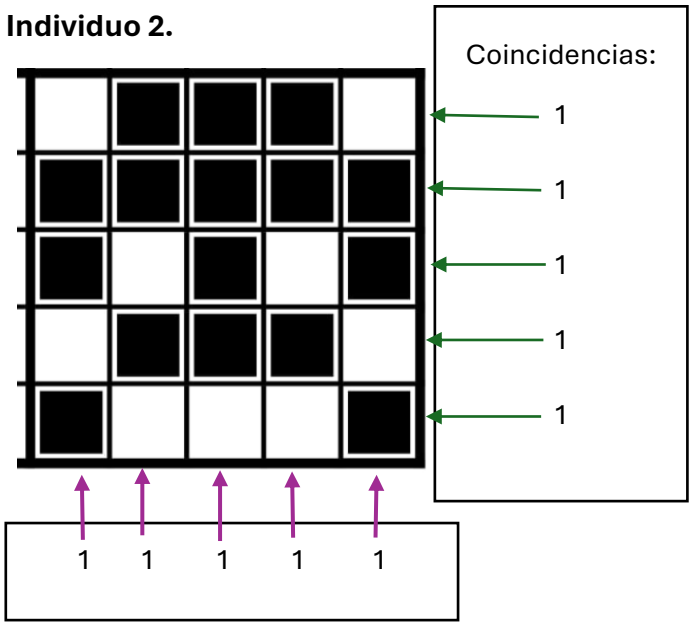
x_c coincidencias en las columnas

Coincidencia: Se cumple la regla de la fila/columna total o parcialmente.

b) Indique el valor de aptitud para los siguientes individuos en una población



El valor de aptitud para la solución planteada por el individuo 1 es de 2.



El valor de aptitud para la solucion planteada por el individuo 2 es de 10, lo cual es igual la suma de la cantidad de filas y columnas, lo que significa que es la solucion perfecta al nonograma.