LAPORAN OBSERVASI UNTUK MENCARI NILAI MINIMUM DARI SUATU FUNGSI MENGGUNAKAN ALGORITMA GENETIKA

Nama : Aditya Gumilar NIM : 1301184037 Kelas : IF-4205

Algoritma Genetik merupakan teknik pencarian didalam computer dalam menemukan penyelesaian perkiraan untuk optimasi dan masalah dalam pencarian. Algoritma Genetika merupakan kelas khusus dari algoritma evolusioner dengan menggunakan teknik yang terinspirasi dari biologi evolusioner seperti warisan, mutasi, cross over, dll.

Diberikan kasus seperti dibawah ini :

$$h(x_1, x_2) = \cos(x_1)\sin(x_2) - \frac{x_1}{(x_2^2 + 1)} \qquad \text{dengan batasan } -1 \le x_1 \le 2 \text{ dan } -1 \le x_2 \le 1.$$

Hasil Observasi

- **Desain Kromosom dan Metode Pendekodean**

	x_1			x_2	
0	1	0	1	1	0

Kromosom yang didesain di Encode menjadi 3 gen menggunakan Binary Encoding 3 bit. Masing-masing dari x1 dan x2 direpresentasikan kedalam 3 bit sehingga ini adalah bentuk Genotype nya. Untuk menghitung Phenotypenya agar mendapatkan nilai x1 dan x2 dapat dihitung menggunakan rumus:

$$x = r_{min} + \frac{r_{max} - r_{min}}{\Sigma_{i=1}^{N} 2^{-i}} (g_1 * 2^{-1} + g_1 * 2^{-2} + \cdots + g_N * 2^{-N})$$

- **Ukuran Populasi**
  Dalam kasus ini saya menggunakan ukuran populasi sebanyak 50 populasi dan max generasinya sebanyak 50. Panjang setiap kromosomnya adalah 10.

- **Pemilihan Orang Tua**
  Metode yang digunakan dalam pemilihan orang tua adalah metode Roulette Wheel. Dalam metode ini kita menghitung probabilitas dari masing-masing individu berdasarkan hasil nilai fitnessnya.

- **Pemilihan dan Teknik Operasi Genetik**
  - Crossover
    Menggunakan persilangan dengan 1 titik potong, dimana inputnya berupa ortu_krom (kromosom orang tua) dengan nilai dari probabilitas crossover = 0.7. Pertama kita keluarkan terlebih dahulu panjang dari kromosomnya, selanjutnya membuat variable child yang merupakan copy dari kromosom orang tua. Dalam melakukan proses crossovernya, kita harus membangkitkan nilai randomnya terlebih dahulu karena tidak semua orang tua memiliki anak.
  - Mutasi
    Inputnya berupa kromosom dan nilai dari probabilitas mutasi = 0.1. Mutasi disini saya menggunakan pergen, jadi kita melakukan looping untuk setiap gen dan membangkitkan bilangan random sepanjang kromosom. Jika bilangan random kurang dari probabilitas mutasi maka akan dilakukan switch.

- **Probabilitas Operasi Genetik (Pc dan Pm)**
  Setelah melakukan beberapa kali percobaan dalam merubah nilai probabilitas crossover (PC) dan probabilitas mutasi (PM),defaultnya nilai PC biasanya berkisar antara 0.6 sampai 0.7 dan PM = 0.1. Berikut hasil percobaannya :
  - Nilai PC = 0.7

```
Generasi= 0 Best Fitness= 1.9961424167495878
Generasi= 2 Best Fitness= 2.0216527429522126
Generasi= 4 Best Fitness= 2.0216527429522126
```

  - Nilai PC = 0.6

```
Generasi= 0 Best Fitness= 1.917099073781922
Generasi= 2 Best Fitness= 2.0161198783747976
Generasi= 4 Best Fitness= 2.0161198783747976
Generasi= 6 Best Fitness= 2.0216527429522126
Generasi= 8 Best Fitness= 2.0216527429522126
Generasi= 10 Best Fitness= 2.0216527429522126
```

  Dapat disimpulkan bahwa jika PC = 0.7 mendapatkan bestfitness saat generasi ke-2, jika PC = 0.6 mendapatkan best fitness saat generasi ke-6.

- **Metode Pergantian Generasi (Seleksi Survivor)**
  Metode yang digunakan adalah Elitisme. Dengan metode ini menyelamatkan kromosom terbaik dari populasi sebelumnya berdasarkan nilai dari fitnessnya. Siapa saja yang memiliki nilai fitness tertinggi, maka akan disimpan di dalam populasi baru. Dalam prosesnya, kita mencari id terbaik dari array max yang ada di Fitness. Untuk mencari kromosom terbaik yaitu copy dari populasi yang memiliki id terbaik. Setelah mendapatkan hasilnya, simpan kedalam populasi baru dimana populasi baru mengeluarkan kromosom terbaik dengan vstack (tumpukan kebawah).

- **Kriteria Penghentian Evolusi**
  Penghentian evolusi akan terjadi ketika populasi dan generasi melebihi ukuran yang telah ditentukan sebelumnya. Dalam kasus ini, saya setting ukuran populasi = 50 dan maxgenerasinya = 50

- **Hasil Output**

```
Generasi= 0 Best Fitness= 1.8025619344561201
Generasi= 2 Best Fitness= 2.0216527429522126
Generasi= 4 Best Fitness= 2.0216527429522126
Generasi= 6 Best Fitness= 2.0216527429522126
Generasi= 8 Best Fitness= 2.0216527429522126
Generasi= 10 Best Fitness= 2.0216527429522126
Generasi= 12 Best Fitness= 2.0216527429522126
Generasi= 14 Best Fitness= 2.0216527429522126
Generasi= 16 Best Fitness= 2.0216527429522126
Generasi= 18 Best Fitness= 2.0216527429522126
Generasi= 20 Best Fitness= 2.0216527429522126
Generasi= 22 Best Fitness= 2.0216527429522126
Generasi= 24 Best Fitness= 2.0216527429522126
Generasi= 26 Best Fitness= 2.0216527429522126
Generasi= 28 Best Fitness= 2.0216527429522126
Generasi= 30 Best Fitness= 2.0216527429522126
Generasi= 32 Best Fitness= 2.0216527429522126
Generasi= 34 Best Fitness= 2.0216527429522126
Generasi= 36 Best Fitness= 2.0216527429522126
Generasi= 38 Best Fitness= 2.0216527429522126
Generasi= 40 Best Fitness= 2.0216527429522126
Generasi= 42 Best Fitness= 2.0216527429522126
Generasi= 44 Best Fitness= 2.0216527429522126
Generasi= 46 Best Fitness= 2.0216527429522126
Generasi= 48 Best Fitness= 2.0216527429522126
```

Hasil output menampilkan setiap generasi genap, karena pada main program terdapat kondisi if dimana jika jumlah generasi habis dibagi 2, maka keluarkan nilainya. Telah dijelaskan tadi bahwa max generasi yang saya setting adalah 50, sehingga pada output ini hanya menampilkan sampai 48.

```
Dekode Terbaik  = [2.         0.09677419]
Fitness Terbaik = 2.0216527429522126
```