In [1]:
```python
import random
```

In [2]:
```python
# Data konfigurasi
HARI = ['Senin', 'Selasa', 'Rabu', 'Kamis', 'Jumat']
PELAJARAN = ['Metodologi Riset dan Penulisan Ilmiah', 'Komunikasi dan Jaringan Komputer', 'Kecerdasan Buatan', 'Pembelajaran d
DOSEN = ['Firman Arifin', 'Iwan Syarif', 'Aliridho Barakbah', 'Riyanto Sigit', 'Riyanto Sigit', 'Tri Budi Santoso']
WAKTU = ['08:00 sd 09:40','09:40 sd 10:30', '10:30 sd 12:10', '13:10 sd 15:40','15:40 sd 16:30']
KELAS = ['PS-03.17', 'PS-08.07',]
```

In [3]:
```python
# Parameter algoritma genetika
POPULASI_SIZE = 50
GENERASI_MAX = 200
MUTASI_RATE = 0.1
```

In [4]:
```python
# Representasi individu (jadwal)
def buat_individu():
    return [
        {
            "dosen": random.choice(DOSEN),
            "kelas": random.choice(KELAS),
            "pelajaran": random.choice(PELAJARAN),
            "waktu": random.choice(WAKTU),
            "hari": random.choice(HARI),
        }
        for _ in range(len(PELAJARAN) * len(KELAS))
        # for _ in range(len(HARI) * len(WAKTU))
    ]
print(len(PELAJARAN), len(KELAS))
```

```
6 2
```

In [5]:
```python
# # Evaluasi fitness (menghitung jumlah konflik)
# def evaluasi_fitness(individu):
#     konflik = 0
#     for i, jadwal1 in enumerate(individu):
#         for j, jadwal2 in enumerate(individu):
#             if i >= j:
#                 continue
```

```
#             # Konflik waktu di kelas yang sama
#             if jadwal1["kelas"] == jadwal2["kelas"] and jadwal1["waktu"] == jadwal2["waktu"]:
#                 konflik += 1
#             # Konflik dosen pada waktu yang sama
#             if jadwal1["dosen"] == jadwal2["dosen"] and jadwal1["waktu"] == jadwal2["waktu"]:
#                 konflik += 1
#     return -konflik
```

In [6]:
```
# Evaluasi fitness (menghitung jumlah konflik)
def evaluasi_fitness(individu):
    konflik = 0
    for i, jadwal1 in enumerate(individu):
        for j, jadwal2 in enumerate(individu):
            if i >= j:
                continue
            # Konflik waktu untuk dosen
            if (jadwal1["dosen"] == jadwal2["dosen"] and
                jadwal1["waktu"] == jadwal2["waktu"] and
                jadwal1["hari"] == jadwal2["hari"]):
                konflik += 1
            # Konflik waktu untuk kelas
            if (jadwal1["kelas"] == jadwal2["kelas"] and
                jadwal1["waktu"] == jadwal2["waktu"] and
                jadwal1["hari"] == jadwal2["hari"]):
                konflik += 1
            # Konflik waktu untuk ruang
            if (jadwal1["kelas"] == jadwal2["kelas"] and
                jadwal1["waktu"] == jadwal2["waktu"] and
                jadwal1["hari"] == jadwal2["hari"]):
                konflik += 1
    return -konflik
```

In [7]:
```
# Seleksi (roulette wheel selection)
def seleksi(populasi):
    fitness_total = sum(evaluasi_fitness(individu) for individu in populasi)
    probabilitas = [
        (evaluasi_fitness(individu) / fitness_total) for individu in populasi
    ]
    return random.choices(populasi, weights=probabilitas, k=2)
```

In [8]:
```python
# Crossover (pertukaran bagian individu)
def crossover(parent1, parent2):
    point = random.randint(1, len(parent1) - 1)
    child1 = parent1[:point] + parent2[point:]
    child2 = parent2[:point] + parent1[point:]
    return child1, child2
```

In [9]:
```python
# Mutasi (modifikasi kecil pada individu)
def mutasi(individu):
    if random.random() < MUTASI_RATE:
        indeks = random.randint(0, len(individu) - 1)
        individu[indeks]["waktu"] = random.choice(WAKTU)
    return individu
```

In [10]:
```python
# Algoritma Genetika
def algoritma_genetika():
    populasi = [buat_individu() for _ in range(POPULASI_SIZE)]
    for generasi in range(GENERASI_MAX):
        populasi = sorted(populasi, key=evaluasi_fitness, reverse=True)
        if evaluasi_fitness(populasi[0]) == 0:  # Solusi optimal ditemukan
            break
        next_gen = []
        while len(next_gen) < POPULASI_SIZE:
            parent1, parent2 = seleksi(populasi)
            child1, child2 = crossover(parent1, parent2)
            next_gen.append(mutasi(child1))
            next_gen.append(mutasi(child2))
        populasi = next_gen
        print(f"Generasi {generasi + 1}: Fitness terbaik = {evaluasi_fitness(populasi[0])}")
    return populasi[0]
```

In [11]:
```python
# Menjalankan algoritma
jadwal_terbaik = algoritma_genetika()
```

In [12]:
```python
# Output jadwal terbaik
for jadwal in jadwal_terbaik:
    print(jadwal)
```

```
{'dosen': 'Tri Budi Santoso', 'kelas': 'PS-03.17', 'pelajaran': 'Kecerdasan Buatan', 'waktu': '10:30 sd 12:10', 'hari': 'Kamis'}
{'dosen': 'Firman Arifin', 'kelas': 'PS-03.17', 'pelajaran': 'Sinyal dan Sistem', 'waktu': '15:40 sd 16:30', 'hari': 'Selasa'}
{'dosen': 'Firman Arifin', 'kelas': 'PS-08.07', 'pelajaran': 'Praktikum Pembelajaran dan Perhitungan Evolusioner', 'waktu': '13:10 sd 15:40', 'hari': 'Selasa'}
{'dosen': 'Riyanto Sigit', 'kelas': 'PS-03.17', 'pelajaran': 'Kecerdasan Buatan', 'waktu': '08:00 sd 09:40', 'hari': 'Selasa'}
{'dosen': 'Riyanto Sigit', 'kelas': 'PS-03.17', 'pelajaran': 'Metodologi Riset dan Penulisan Ilmiah', 'waktu': '08:00 sd 09:40', 'hari': 'Senin'}
{'dosen': 'Firman Arifin', 'kelas': 'PS-08.07', 'pelajaran': 'Pembelajaran dan Perhitungan Evolusioner', 'waktu': '08:00 sd 09:40', 'hari': 'Rabu'}
{'dosen': 'Iwan Syarif', 'kelas': 'PS-08.07', 'pelajaran': 'Kecerdasan Buatan', 'waktu': '08:00 sd 09:40', 'hari': 'Jumat'}
{'dosen': 'Aliridho Barakbah', 'kelas': 'PS-03.17', 'pelajaran': 'Kecerdasan Buatan', 'waktu': '10:30 sd 12:10', 'hari': 'Senin'}
{'dosen': 'Riyanto Sigit', 'kelas': 'PS-03.17', 'pelajaran': 'Metodologi Riset dan Penulisan Ilmiah', 'waktu': '09:40 sd 10:30', 'hari': 'Selasa'}
{'dosen': 'Firman Arifin', 'kelas': 'PS-08.07', 'pelajaran': 'Kecerdasan Buatan', 'waktu': '09:40 sd 10:30', 'hari': 'Kamis'}
{'dosen': 'Aliridho Barakbah', 'kelas': 'PS-08.07', 'pelajaran': 'Komunikasi dan Jaringan Komputer', 'waktu': '13:10 sd 15:40', 'hari': 'Kamis'}
{'dosen': 'Riyanto Sigit', 'kelas': 'PS-08.07', 'pelajaran': 'Pembelajaran dan Perhitungan Evolusioner', 'waktu': '09:40 sd 10:30', 'hari': 'Rabu'}
```

In [13]:

```python
import pandas as pd

# Mengonversi ke DataFrame untuk tabel
df = pd.DataFrame(jadwal_terbaik)
sorted_df = df.sort_values(by=['hari', 'waktu', 'kelas'])
# Output sebagai tabel
print(sorted_df.to_string(index=False))
```

dosen	kelas		pelajaran	waktu	hari
Iwan Syarif	PS-08.07		Kecerdasan Buatan	08:00 sd 09:40	Jumat
Firman Arifin	PS-08.07		Kecerdasan Buatan	09:40 sd 10:30	Kamis
Tri Budi Santoso	PS-03.17		Kecerdasan Buatan	10:30 sd 12:10	Kamis
Aliridho Barakbah	PS-08.07		Komunikasi dan Jaringan Komputer	13:10 sd 15:40	Kamis
Firman Arifin	PS-08.07		Pembelajaran dan Perhitungan Evolusioner	08:00 sd 09:40	Rabu
Riyanto Sigit	PS-08.07		Pembelajaran dan Perhitungan Evolusioner	09:40 sd 10:30	Rabu
Riyanto Sigit	PS-03.17		Kecerdasan Buatan	08:00 sd 09:40	Selasa
Riyanto Sigit	PS-03.17		Metodologi Riset dan Penulisan Ilmiah	09:40 sd 10:30	Selasa
Firman Arifin	PS-08.07	Praktikum	Pembelajaran dan Perhitungan Evolusioner	13:10 sd 15:40	Selasa
Firman Arifin	PS-03.17		Sinyal dan Sistem	15:40 sd 16:30	Selasa
Riyanto Sigit	PS-03.17		Metodologi Riset dan Penulisan Ilmiah	08:00 sd 09:40	Senin
Aliridho Barakbah	PS-03.17		Kecerdasan Buatan	10:30 sd 12:10	Senin