BIS LAB 4

17/10/2025

Sharada Koundinya (1BM23CS310)

CUCKOO SEARCH ALGORITHM:

CODE:

```
import random

tasks = [2, 3, 4, 5, 6]
num_tasks = len(tasks)
num_nests = 5
Pa = 0.25
MaxGen = 50

def fitness(schedule):
    """Lower total duration = better fitness."""
    total_time = 0
    for t in schedule:
        total_time += t
    return -total_time

def random_schedule():
    """Create a random schedule (random order of tasks)."""
    s = tasks[:]
    random.shuffle(s)
    return s

def levy_flight(schedule):
    """Generate new schedule by small random changes."""
    new_s = schedule[:]
    i, j = random.sample(range(num_tasks), 2)
    new_s[i], new_s[j] = new_s[j], new_s[i]
    return new_s

nests = [random_schedule() for _ in range(num_nests)]
fitness_values = [fitness(s) for s in nests]

for gen in range(MaxGen):
    for i in range(num_nests):
        new_solution = levy_flight(nests[i])
        new_fitness = fitness(new_solution)

        j = random.randint(0, num_nests - 1)
        if new_fitness > fitness_values[j]:
            nests[j] = new_solution
            fitness_values[j] = new_fitness
```

```
    sorted_nests = sorted(zip(fitness_values, nests), reverse=True)
    num_abandon = int(Pa * num_nests)
    for k in range(num_abandon):
        sorted_nests[-(k+1)] = (fitness(random_schedule()), random_schedule())

    fitness_values, nests = zip(*sorted_nests)
    fitness_values, nests = list(fitness_values), list(nests)

best_index = fitness_values.index(max(fitness_values))
print("Best Schedule:", nests[best_index])
print("Total Time:", -fitness_values[best_index])
```

OUTPUT:

```
Best Schedule: [6, 5, 4, 3, 2]
Total Time: 20
```