```
import random
import math  # cos() for Rastrigin
import copy  # array-copying convenience
import sys   # max float

# -------fitness functions---------

# Rastrigin function
def fitness_rastrigin(position):
    fitnessVal = 0.0
    for i in range(len(position)):
        xi = position[i]
        fitnessVal += (xi * xi) - (10 * math.cos(2 * math.pi * xi)) + 10
    return fitnessVal

# Sphere function
def fitness_sphere(position):
    fitnessVal = 0.0
    for i in range(len(position)):
        xi = position[i]
        fitnessVal += (xi * xi)
    return fitnessVal

# -------------------------

# Particle class
class Particle:
    def __init__(self, fitness, dim, minx, maxx, seed):
        self.rnd = random.Random(seed)

        # Initialize position of the particle
        self.position = [0.0 for i in range(dim)]

        # Initialize velocity of the particle
        self.velocity = [0.0 for i in range(dim)]

        # Initialize best particle position
        self.best_part_pos = [0.0 for i in range(dim)]

        # Initialize random position and velocity
        for i in range(dim):
            self.position[i] = ((maxx - minx) * self.rnd.random() + minx)
            self.velocity[i] = ((maxx - minx) * self.rnd.random() + minx)

        # Compute fitness of particle
        self.fitness = fitness(self.position)

        # Initialize best position and fitness
        self.best_part_pos = copy.copy(self.position)
        self.best_part_fitnessVal = self.fitness

# Particle Swarm Optimization function
def pso(fitness, max_iter, n, dim, minx, maxx):
    # Hyperparameters
    w = 0.729    # inertia
    c1 = 1.49445  # cognitive (particle)
    c2 = 1.49445  # social (swarm)

    rnd = random.Random(0)

    # Create n random particles
    swarm = [Particle(fitness, dim, minx, maxx, i) for i in range(n)]

    # Initialize the best position and fitness in the swarm
    best_swarm_pos = [0.0 for i in range(dim)]
    best_swarm_fitnessVal = sys.float_info.max

    # Compute best particle in the swarm
    for i in range(n):
        if swarm[i].fitness < best_swarm_fitnessVal:
            best_swarm_fitnessVal = swarm[i].fitness
            best_swarm_pos = copy.copy(swarm[i].position)

    # Main loop of PSO
    Iter = 0
    while Iter < max_iter:

        # Print iteration info
        if Iter % 10 == 0 and Iter > 1:
            print(f"Iter = {Iter} best fitness = {best_swarm_fitnessVal:.3f}")
```

```
            for i in range(n):
                # Update velocity
                for k in range(dim):
                    r1 = rnd.random()
                    r2 = rnd.random()

                    swarm[i].velocity[k] = (
                        (w * swarm[i].velocity[k]) +
                        (c1 * r1 * (swarm[i].best_part_pos[k] - swarm[i].position[k])) +
                        (c2 * r2 * (best_swarm_pos[k] - swarm[i].position[k]))
                    )

                    # Clip velocity to within bounds
                    if swarm[i].velocity[k] < minx:
                        swarm[i].velocity[k] = minx
                    elif swarm[i].velocity[k] > maxx:
                        swarm[i].velocity[k] = maxx

                # Update position
                for k in range(dim):
                    swarm[i].position[k] += swarm[i].velocity[k]

                # Update fitness
                swarm[i].fitness = fitness(swarm[i].position)

                # Update personal best
                if swarm[i].fitness < swarm[i].best_part_fitnessVal:
                    swarm[i].best_part_fitnessVal = swarm[i].fitness
                    swarm[i].best_part_pos = copy.copy(swarm[i].position)

                # Update global best
                if swarm[i].fitness < best_swarm_fitnessVal:
                    best_swarm_fitnessVal = swarm[i].fitness
                    best_swarm_pos = copy.copy(swarm[i].position)

            Iter += 1

        return best_swarm_pos

    # ----------------------------
    # Driver code for Rastrigin function

    print("\nBegin particle swarm optimization on Rastrigin function\n")
    dim = 3
    fitness = fitness_rastrigin

    print(f"Goal is to minimize Rastrigin's function in {dim} variables")
    print("Function has known min = 0.0 at (", end="")
    for i in range(dim - 1):
        print("0, ", end="")
    print("0)")

    num_particles = 50
    max_iter = 100

    print(f"Setting num_particles = {num_particles}")
    print(f"Setting max_iter = {max_iter}")
    print("\nStarting PSO algorithm\n")

    best_position = pso(fitness, max_iter, num_particles, dim, -10.0, 10.0)

    print("\nPSO completed\n")
    print("\nBest solution found:")
    print([f"{best_position[k]:.6f}" for k in range(dim)])
    fitnessVal = fitness(best_position)
    print(f"Fitness of best solution = {fitnessVal:.6f}")

    print("\nEnd particle swarm for Rastrigin function\n")
    print()

    # Driver code for Sphere function
    print("\nBegin particle swarm optimization on Sphere function\n")
    dim = 3
    fitness = fitness_sphere

    print(f"Goal is to minimize Sphere function in {dim} variables")
    print("Function has known min = 0.0 at (", end="")
    for i in range(dim - 1):
        print("0, ", end="")
    print("0)")

    print(f"Setting num_particles = {num_particles}")
    print(f"Setting max_iter = {max_iter}")
```

```
print("\nStarting PSO algorithm\n")

best_position = pso(fitness, max_iter, num_particles, dim, -10.0, 10.0)

print("\nPSO completed\n")
print("\nBest solution found:")
print([f"{best_position[k]:.6f}" for k in range(dim)])
fitnessVal = fitness(best_position)
print(f"Fitness of best solution = {fitnessVal:.6f}")

print("\nEnd particle swarm for Sphere function\n")
```

```
Begin particle swarm optimization on Rastrigin function

Goal is to minimize Rastrigin's function in 3 variables
Function has known min = 0.0 at (0, 0, 0)
Setting num_particles = 50
Setting max_iter = 100

Starting PSO algorithm

Iter = 10 best fitness = 8.463
Iter = 20 best fitness = 4.792
Iter = 30 best fitness = 2.223
Iter = 40 best fitness = 0.251
Iter = 50 best fitness = 0.251
Iter = 60 best fitness = 0.061
Iter = 70 best fitness = 0.007
Iter = 80 best fitness = 0.005
Iter = 90 best fitness = 0.000

PSO completed


Best solution found:
['0.000618', '0.000013', '0.000616']
Fitness of best solution = 0.000151

End particle swarm for Rastrigin function



Begin particle swarm optimization on Sphere function

Goal is to minimize Sphere function in 3 variables
Function has known min = 0.0 at (0, 0, 0)
Setting num_particles = 50
Setting max_iter = 100

Starting PSO algorithm

Iter = 10 best fitness = 0.189
Iter = 20 best fitness = 0.012
Iter = 30 best fitness = 0.001
Iter = 40 best fitness = 0.000
Iter = 50 best fitness = 0.000
Iter = 60 best fitness = 0.000
Iter = 70 best fitness = 0.000
Iter = 80 best fitness = 0.000
Iter = 90 best fitness = 0.000

PSO completed


Best solution found:
['0.000004', '-0.000001', '0.000007']
Fitness of best solution = 0.000000

End particle swarm for Sphere function
```