```python
#!/usr/bin/python3.4
# -*-coding:Utf-8 -*

from lib.inject_arguments import inject_arguments


class GameOptimizer:
    @inject_arguments
    def __init__(self, event_dispatcher):
        self.event_dispatcher.listen('game.frame', self.onFrame)
        self.event_dispatcher.listen('action', self.onAction)


    def onFrame(self, frame):
        # Reset
        if frame.current_frame < 5:
            self.action_detected = False
            self.mario_x = 0
            self.last_mario_x_change = 0
            self.last_points = []

        # Detect inactivity (10 frames)
        if frame.current_frame > 10 and not self.action_detected:
            self.event_dispatcher.dispatch('stop')

        # Detect x-inactive IA (2,5 sec == 150 frames)
        if self.mario_x != frame.mario.rect.x:
            self.last_mario_x_change = frame.current_frame
        if frame.current_frame > self.last_mario_x_change + 150:
            self.event_dispatcher.dispatch('stop')

        self.mario_x = frame.mario.rect.x

        # Detect looping IA (12 sec == 720 frames)
        point = int(self.mario_x / 10), int(frame.mario.rect.y / 10)
        self.last_points.append(point)
        if len(self.last_points) > 720:
            self.last_points.pop(0)

        if frame.current_frame < 720:
            return
        indexes = [i for i, v in enumerate(self.last_points) if v == point]
        if len(indexes) >= 5:
            self.event_dispatcher.dispatch('stop')


    def onAction(self, event):
        self.action_detected = True
```