**Лабораторная работа 3**

**Задание.**

Написать простенькое приложение "Барсик" - который умеет прыгать по оординатам с погрешностью (во время прыжка худеет -0,2 кг) - который умеет есть (+1,5 кг). ,Барсика выпускают на дачу, он весь день бесится (произвольно или по команде пользователя - ест и прыгает во всех сочетаниях). По окончании дня выяснить, где голод или вечер застали зверушку и в каком радиусе его исать (наопленная погрешность прышков).

**Решение.**

**Код:**

import Foundation

class Barsik {

var weight: Double

var position: (x: Double, y: Double)

var totalDistance: Double

var hungerLevel: Int // 0 - сытый, 10 - очень голодный

init(weight: Double, startPosition: (x: Double, y: Double)) {

self.weight = weight

self.position = startPosition

self.totalDistance = 0.0

self.hungerLevel = 0

}

func jump(distance: Double, angle: Double) {

guard weight > 0.3 else {

print("Барсик слишком худой и не может прыгать! 😿")

return

}

let errorFactor = Double(hungerLevel) \* 0.1 + 0.05

let actualDistance = distance \* (1.0 + Double.random(in: -errorFactor...errorFactor))

let actualAngle = angle + Double.random(in: -0.2...0.2)

let deltaX = actualDistance \* cos(actualAngle)

let deltaY = actualDistance \* sin(actualAngle)

position.x += deltaX

position.y += deltaY

totalDistance += actualDistance

weight -= 0.2

hungerLevel += 1

print("Барсик прыгнул на \(String(format: "%.2f", actualDistance)) метров!")

print("Новые координаты: (\(String(format: "%.2f", position.x)), \(String(format: "%.2f", position.y)))")

print("Вес уменьшился до: \(String(format: "%.1f", weight)) кг")

print("Уровень голода: \(hungerLevel)/10")

print("---")

}

func eat() {

weight += 1.5

hungerLevel = max(0, hungerLevel - 3)

print("Барсик поел! 🍗")

print("Вес увеличился до: \(String(format: "%.1f", weight)) кг")

print("Уровень голода: \(hungerLevel)/10")

print("---")

}

func randomAction() {

let actions = ["jump", "eat", "jump", "jump"] // Чаще прыгает

switch actions.randomElement() {

case "jump":

let randomDistance = Double.random(in: 1.0...5.0)

let randomAngle = Double.random(in: 0...2 \* .pi)

jump(distance: randomDistance, angle: randomAngle)

case "eat":

if hungerLevel >= 2 { // Ест только когда достаточно голоден

eat()

} else {

// Если не голоден, вместо еды прыгает

let randomDistance = Double.random(in: 1.0...3.0)

let randomAngle = Double.random(in: 0...2 \* .pi)

jump(distance: randomDistance, angle: randomAngle)

}

default:

break

}

}

func printStatus() {

print("=== ТЕКУЩЕЕ СОСТОЯНИЕ БАРСИКА ===")

print("Вес: \(String(format: "%.1f", weight)) кг")

print("Координаты: (\(String(format: "%.2f", position.x)), \(String(format: "%.2f", position.y)))")

print("Уровень голода: \(hungerLevel)/10")

print("Общая дистанция прыжков: \(String(format: "%.2f", totalDistance)) метров")

print("Радиус поиска: \(String(format: "%.2f", calculateSearchRadius())) метров")

print("---")

}

func calculateSearchRadius() -> Double {

// Радиус поиска зависит от накопленной погрешности и голода

return totalDistance \* 0.15 + Double(hungerLevel) \* 0.5

}

func finalReport() {

print("\n🎯 === ИТОГ ДНЯ ===")

print("Барсик закончил день в точке: (\(String(format: "%.2f", position.x)), \(String(format: "%.2f", position.y)))")

print("Финальный вес: \(String(format: "%.1f", weight)) кг")

let hungerStatus: String

switch hungerLevel {

case 0...2:

hungerStatus = "сытый и довольный 😸"

case 3...5:

hungerStatus = "немного голоден 🐈"

case 6...8:

hungerStatus = "голодный 😾"

default:

hungerStatus = "очень голодный! Нужно срочно кормить! 🙀"

}

print("Состояние: \(hungerStatus)")

let searchRadius = calculateSearchRadius()

print("Ищите Барсика в радиусе \(String(format: "%.2f", searchRadius)) метров от начальной точки")

let distanceFromStart = sqrt(pow(position.x, 2) + pow(position.y, 2))

print("Текущее расстояние от дома: \(String(format: "%.2f", distanceFromStart)) метров")

}

}

func main() {

print("🐱 ДОБРО ПОЖАЛОВАТЬ В СИМУЛЯТОР БАРСИКА!")

print("Барсик выпущен на дачу...\n")

let barsik = Barsik(weight: 5.0, startPosition: (x: 0, y: 0))

var dayEnded = false

while !dayEnded {

print("Выберите действие:")

print("1 - Барсик бесится сам")

print("2 - Барсик прыгает")

print("3 - Барсик ест")

print("4 - Показать статус")

print("5 - Завершить день")

print("Введите номер действия:", terminator: " ")

if let input = readLine(), let choice = Int(input) {

switch choice {

case 1:

print("\nБарсик бесится! 🎉")

for \_ in 1...3 {

barsik.randomAction()

// Небольшая пауза между действиями

Thread.sleep(forTimeInterval: 0.5)

}

case 2:

print("\nВведите дистанцию прыжка:", terminator: " ")

if let distInput = readLine(), let distance = Double(distInput) {

print("Введите угол (в радианах, 0-6.28):", terminator: " ")

if let angleInput = readLine(), let angle = Double(angleInput) {

barsik.jump(distance: distance, angle: angle)

}

}

case 3:

barsik.eat()

case 4:

barsik.printStatus()

case 5:

dayEnded = true

barsik.finalReport()

default:

print("Неверный выбор! Попробуйте снова.")

}

} else {

print("Пожалуйста, введите число от 1 до 5")

}

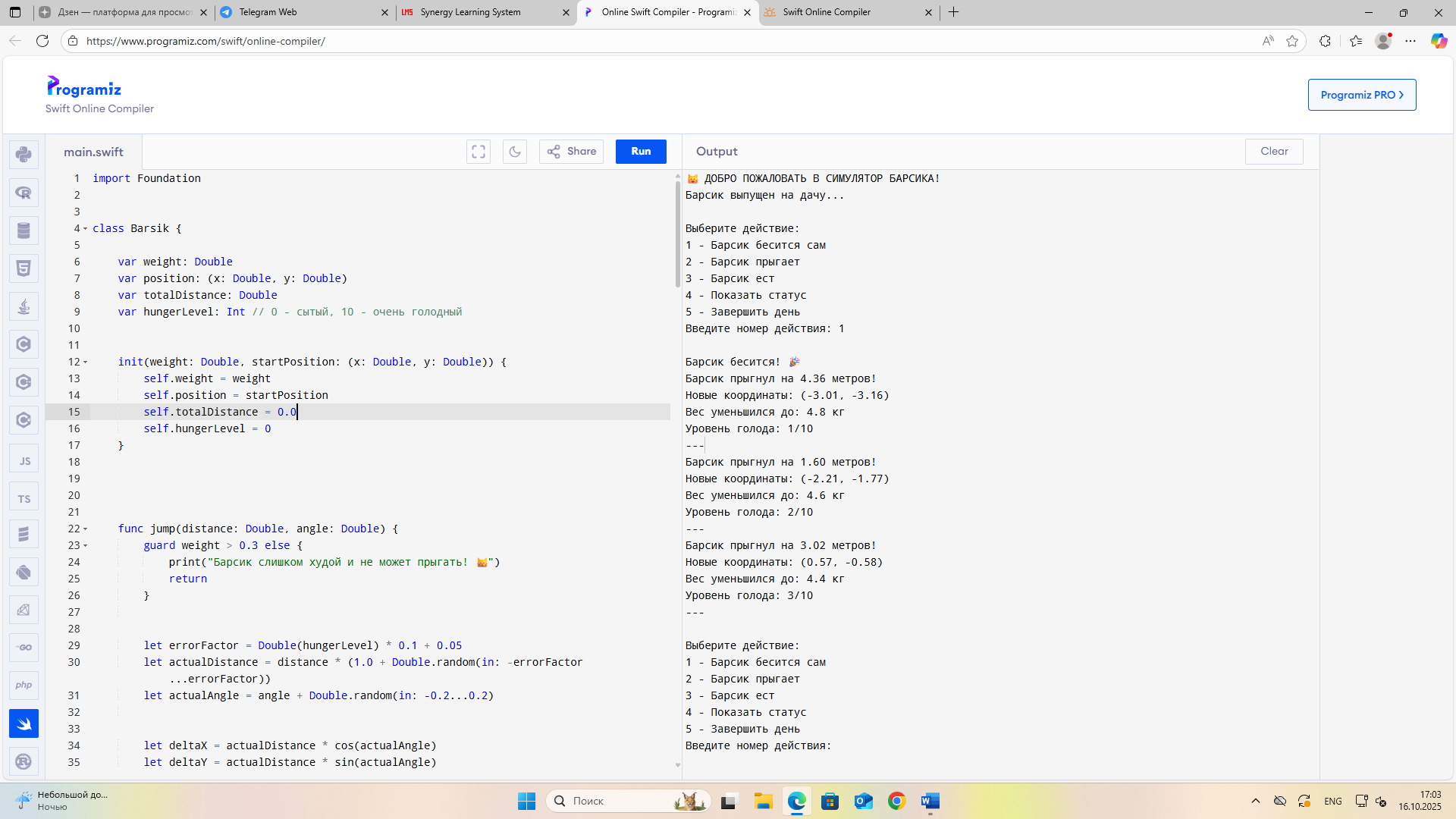
print()

}

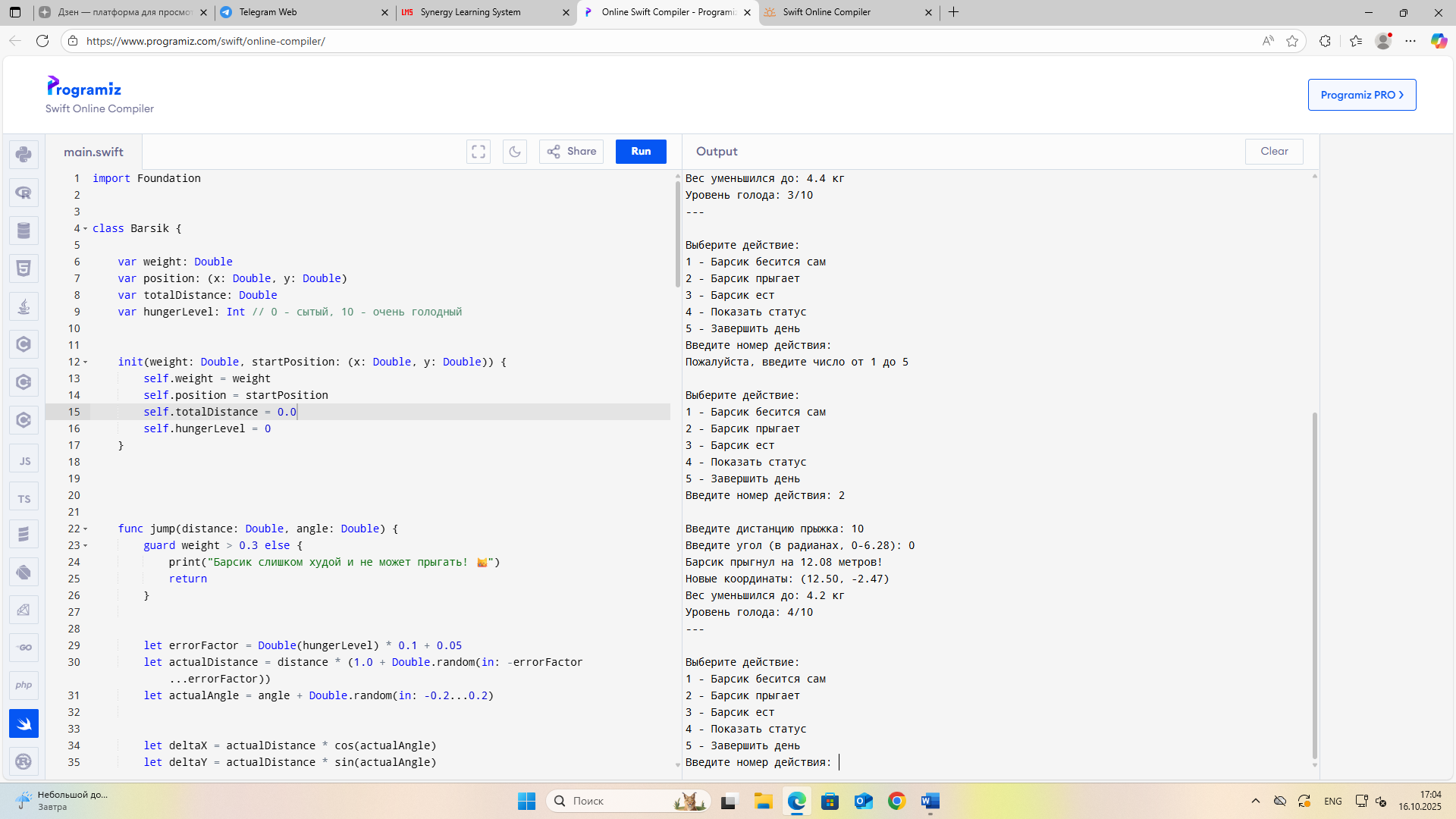
}

main()

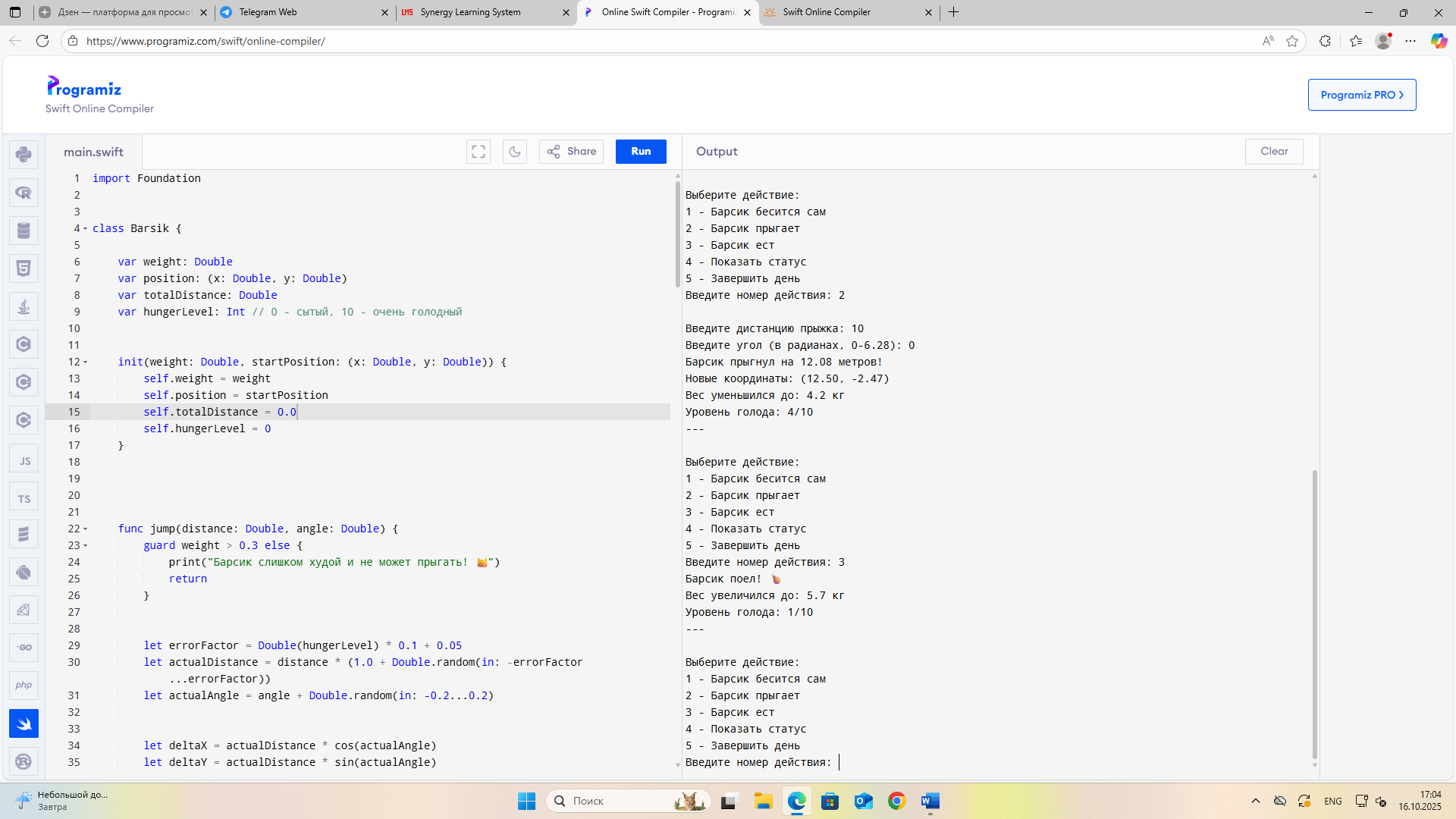
Скриншоты как работает:



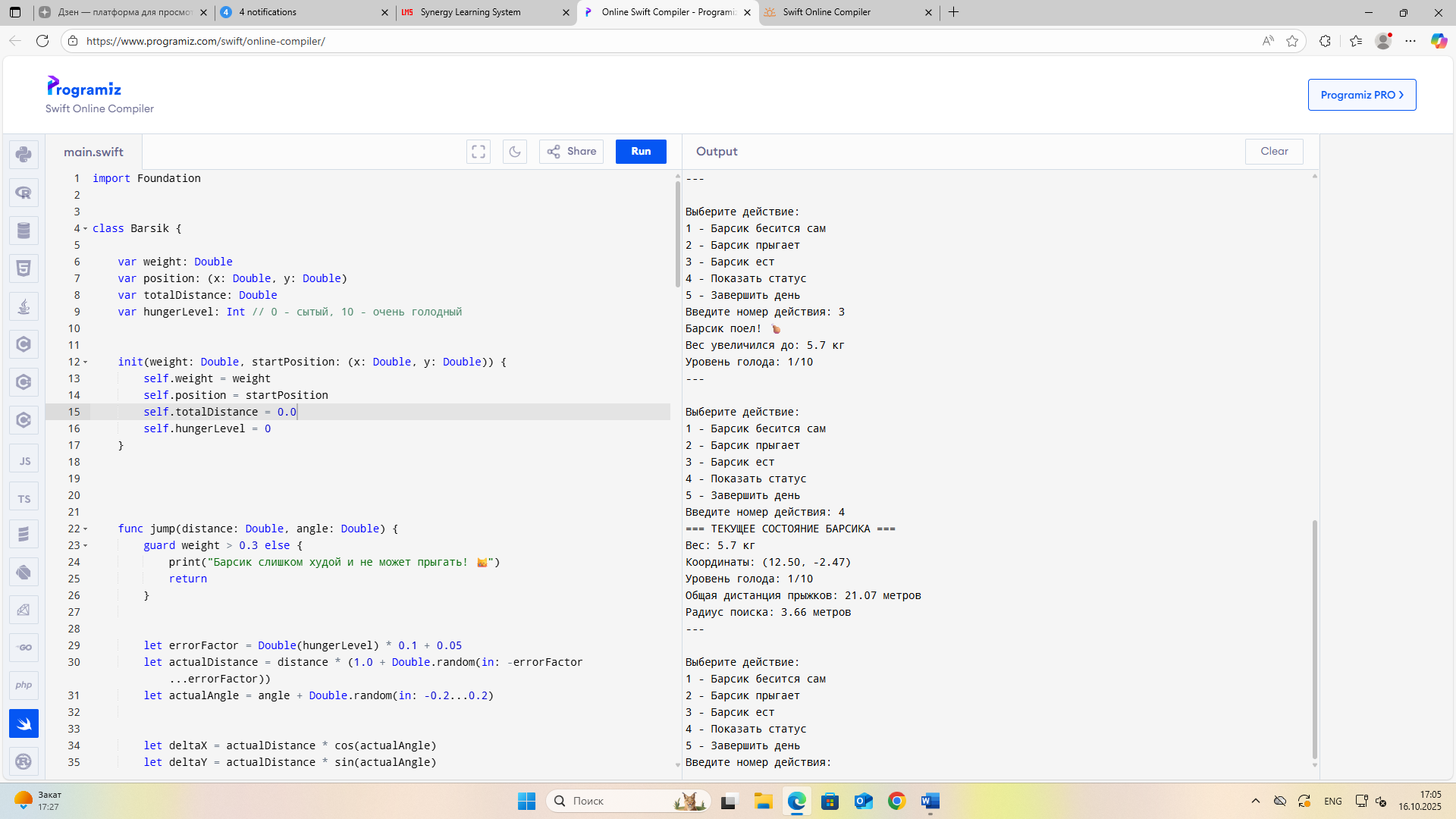
1 – Барсик прыгает сам



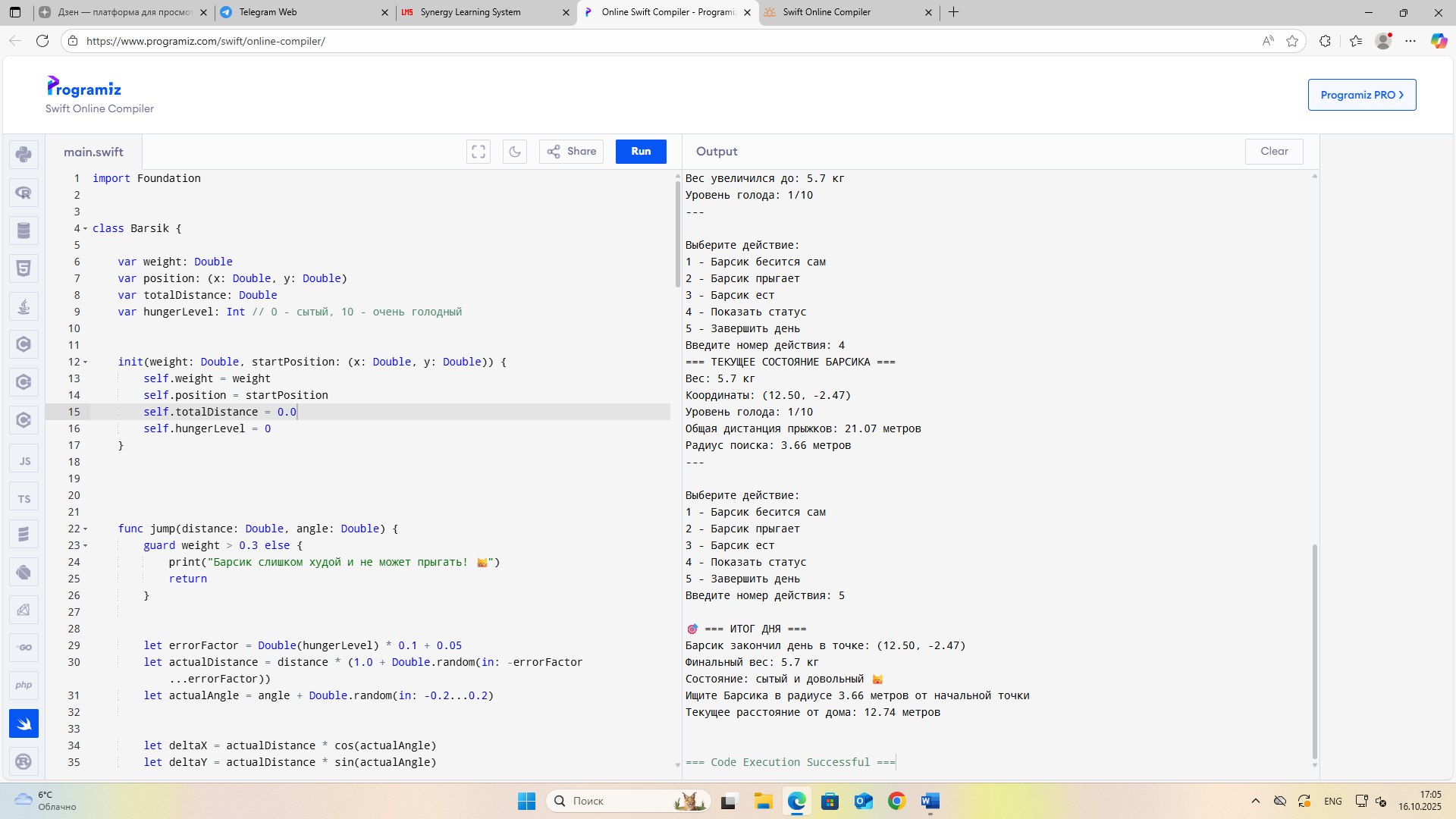
2 – Барсик прыгает



3 – барсик ест



4 – статус барсика



5 – Завершить день