7.4.8

EE25BTECH11001 - Aarush Dilawri

October 11, 2025

Question:

For each natural number k, let C_k denote the circle with radius k centimetres and centre at the origin. On the circle C_k , a particle moves k centimetres in the counter- clockwise direction. After completing its motion on C_k , the particle moves to C_{k+1} in the radial direction. The motion of the particle continues in this manner. The particle starts at $(1, 0)$. If the particle crosses the positive direction of the X axis for the first time on the Circle C_n, then $n =$ _

Solution:

$$\text{Let} \quad \mathbf{p}_0 = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \tag{1}$$

We model a rotation by an angle θ using the rotation matrix

$$R(\theta) = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \tag{2}$$

Note the group property of rotations:

$$R(\theta_1)\, R(\theta_2) = R(\theta_1 + \theta_2), \qquad R(\theta)^k = R(k\theta). \tag{3}$$

Solution

On the circle C_k the particle moves an arc of length k on a circle of radius k, so the angular increment on C_k is

$$\Delta\theta_k = \frac{\text{arc length}}{\text{radius}} = \frac{k}{k} = 1 \quad \text{(radian).} \tag{4}$$

Thus each circular motion rotates the particle by 1 radian. We track the position of the particle at the instant it finishes its motion on C_k (that is, after the arc motion but before the radial jump to C_{k+1}).

Solution

Starting at $\mathbf{p}_0$ on C_1, after finishing C_1 the position is

$$\mathbf{P}_1 = 1\, R(1)\, \mathbf{p}_0. \tag{5}$$

Then the particle moves radially to C_2, scaling the radius from 1 to 2, so just before moving on C_2 the vector is $2R(1)\mathbf{p}_0$. After moving on C_2 (an additional rotation by 1) the particle is at

$$\mathbf{P}_2 = 2\, R(1)R(1)\, \mathbf{p}_0 \;=\; 2\, R(2)\, \mathbf{p}_0. \tag{6}$$

By induction, after finishing its motion on C_k the particle is at

$$\mathbf{P}_k = k\, R(k)\, \mathbf{p}_0. \tag{7}$$

Solution

Therefore the angular coordinate of the particle after completing C_k is exactly k radians. The motion on C_n runs the angle from $(n-1)$ to n (radians). Hence the particle crosses the positive x-axis during the motion on C_n precisely when some integer multiple of 2π lies in the interval $(n-1, n]$, i.e. when there exists $m \in \mathbb{N}$ such that

$$n - 1 \; < \; 2\pi m \; \leq \; n. \tag{8}$$

Solution

We look for the smallest natural number n for which this happens. Take $m = 1$ (the first positive multiple of 2π). Compute

$$2\pi \approx 6.283185307\ldots \tag{9}$$

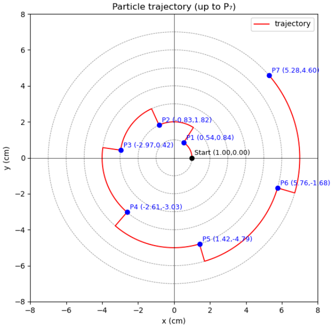
and observe

$$6 \; < \; 2\pi \; \le \; 7. \tag{10}$$

Thus 2π lies in the interval $(6, 7]$, so the condition holds for $n = 7$ (with $m = 1$). For any $n \le 6$ the interval $(n-1, n]$ is contained in $[0, 6]$ and cannot contain $2\pi \approx 6.283\ldots$.
Therefore the particle crosses the positive x-axis for the first time while moving on C_n with

$$\boxed{n \; = \; 7.} \tag{11}$$

Graphical Representation



C Code (code.c)

```c
#include <stdio.h>
#include <math.h>

void particle_endpoints(int n, double *px, double *py, double *theta_out
    ) {
    double theta = 0.0;
    for (int k = 1; k <= n; ++k) {
        double r = (double)k;
        // Arc length on C_k = k   = 1 rad
        double delta = 1.0;
        theta += delta;
        px[k-1] = r * cos(theta);
        py[k-1] = r * sin(theta);
        if (theta_out != NULL) theta_out[k-1] = theta;
    }
}
```

Python Code (code.py)

```
import numpy as np
import matplotlib.pyplot as plt

# ----- Parameters
    ------------------------------------------------

n = 7
arc_samples = 120
radial_samples = 30

# ----- Compute endpoints and angles
    ------------------------------------------------

px, py, thetas = np.zeros(n), np.zeros(n), np.zeros(n)
theta = 0.0
for k in range(1, n + 1):
    theta += 1.0 # 1 rad per circle
```

Python Code (code.py)

```
path_x, path_y = [], []
theta_prev = 0.0
r_prev = 1.0
path_x.append(r_prev * np.cos(theta_prev))
path_y.append(r_prev * np.sin(theta_prev))

for k in range(1, n + 1):
    theta_curr = float(thetas[k - 1])
    r_curr = float(k)

    if k == 1:
        # Arc on C1
        angles = np.linspace(theta_prev, theta_curr, arc_samples + 1)[1:]
        for ang in angles:
            path_x.append(r_curr * np.cos(ang))
            path_y.append(r_curr * np.sin(ang))
```

Python Code (code.py)

```
else:
    # Radial outward
    radii = np.linspace(r_prev, r_curr, radial_samples + 1)[1:]
    for rad in radii:
        path_x.append(rad * np.cos(theta_prev))
        path_y.append(rad * np.sin(theta_prev))

    # Arc on C_k
    angles = np.linspace(theta_prev, theta_curr, arc_samples + 1)[1:]
    for ang in angles:
        path_x.append(r_curr * np.cos(ang))
        path_y.append(r_curr * np.sin(ang))

theta_prev, r_prev = theta_curr, r_curr
```

Python Code (code.py)

```
fig, ax = plt.subplots(figsize=(7, 7))
ax.set_aspect("equal")
ax.set_title("Particle trajectory (pure Python)")

# Circles
theta_full = np.linspace(0, 2 * np.pi, 400)
for k in range(1, n + 1):
    ax.plot(k * np.cos(theta_full), k * np.sin(theta_full),
            linestyle="--", color="gray", linewidth=0.6)

# Trajectory
ax.plot(path_x, path_y, color="red", linewidth=1.3, label="trajectory")

# Points P..P with coordinates
ax.scatter(px, py, color="blue", zorder=5)
```

Python Code (code.py)

```
for i in range(n):
    label = f"P{i+1} ({px[i]:.2f},{py[i]:.2f})"
    ax.text(px[i] + 0.15, py[i] + 0.15, label, fontsize=9, color="blue")
ax.scatter([1.0], [0.0], color="black", zorder=6)
ax.text(1.0 + 0.15, 0.0 + 0.15, "Start (1.00,0.00)", fontsize=9, color="
    black")
ax.axhline(0, color="k", linewidth=0.5)
ax.axvline(0, color="k", linewidth=0.5)
lim = n + 1
ax.set_xlim(-lim, lim)
ax.set_ylim(-lim, lim)
ax.set_xlabel("x (cm)")
ax.set_ylabel("y (cm)")
ax.legend()
plt.show()
```

Python Code (nativecode.py)

```
import ctypes
import numpy as np
import matplotlib.pyplot as plt
lib = ctypes.CDLL("./code.so")
lib.particle_endpoints.argtypes = [
    ctypes.c_int,
    np.ctypeslib.ndpointer(dtype=np.double, ndim=1, flags="
        C_CONTIGUOUS"),
    np.ctypeslib.ndpointer(dtype=np.double, ndim=1, flags="
        C_CONTIGUOUS"),
    np.ctypeslib.ndpointer(dtype=np.double, ndim=1, flags="
        C_CONTIGUOUS")
]
lib.particle_endpoints.restype = None
n = 7
arc_samples = 120
radial_samples = 30
```

Python Code (nativecode.py)

```
px = np.zeros(n, dtype=np.double)
py = np.zeros(n, dtype=np.double)
thetas = np.zeros(n, dtype=np.double)
lib.particle_endpoints(n, px, py, thetas)

# ----- Build continuous trajectory
    -----------------------------------------

path_x, path_y = [], []
theta_prev = 0.0
r_prev = 1.0
path_x.append(r_prev * np.cos(theta_prev))
path_y.append(r_prev * np.sin(theta_prev))

for k in range(1, n + 1):
    theta_curr = float(thetas[k - 1])
    r_curr = float(k)
```

Python Code (nativecode.py)

```
if k == 1:
    # Only arc on C1
    angles = np.linspace(theta_prev, theta_curr, arc_samples + 1)[1:]
    for ang in angles:
        path_x.append(r_curr * np.cos(ang))
        path_y.append(r_curr * np.sin(ang))
else:
    # Radial outward line
    radii = np.linspace(r_prev, r_curr, radial_samples + 1)[1:]
    for rad in radii:
        path_x.append(rad * np.cos(theta_prev))
        path_y.append(rad * np.sin(theta_prev))

    # Arc motion on C_k
```

Python Code (nativecode.py)

```
        angles = np.linspace(theta_prev, theta_curr, arc_samples + 1)[1:]
        for ang in angles:
            path_x.append(r_curr * np.cos(ang))
            path_y.append(r_curr * np.sin(ang))

    theta_prev, r_prev = theta_curr, r_curr

# ----- Plotting
    -------------------------------------------

fig, ax = plt.subplots(figsize=(7, 7))
ax.set_aspect("equal")
ax.set_title("Particle trajectory (up to P)")

# Draw circles C..C
theta_full = np.linspace(0, 2 * np.pi, 400)
```

Python Code (nativecode.py)

```
for k in range(1, n + 1):
    cx = k * np.cos(theta_full)
    cy = k * np.sin(theta_full)
    ax.plot(cx, cy, linestyle="--", color="gray", linewidth=0.6)

# Plot trajectory
ax.plot(path_x, path_y, color="red", linewidth=1.3, label="trajectory")

# Points P..P with coordinates
ax.scatter(px, py, color="blue", zorder=5)
for i in range(n):
    label = f"P{i+1}-({px[i]:.2f},{py[i]:.2f})"
    ax.text(px[i] + 0.15, py[i] + 0.15, label, fontsize=9, color="blue")
```

Python Code (nativecode.py)

```
ax.scatter([1.0], [0.0], color="black", zorder=6)
ax.text(1.0 + 0.15, 0.0 + 0.15, "Start (1.00,0.00)", fontsize=9, color="
    black")

# Axes and formatting
ax.axhline(0, color="k", linewidth=0.5)
ax.axvline(0, color="k", linewidth=0.5)
lim = n + 1
ax.set_xlim(-lim, lim)
ax.set_ylim(-lim, lim)
ax.set_xlabel("x (cm)")
ax.set_ylabel("y (cm)")
ax.legend()
plt.show()
```