import random

# Define the ITstudent class.

class ITstudent:

    def \_\_init\_\_(self, student\_name, student\_id, programme, courses, marks):

        self.student\_name = student\_name

        self.student\_id = student\_id

        self.programme = programme

        self.courses = courses

        self.marks = marks

    # Calculate the average mark.

    def calculate\_average\_mark(self):

        total\_marks = 0

        for mark in self.marks:

            total\_marks += mark

        average\_mark = total\_marks / len(self.marks)

        return average\_mark

    # Determine whether the student passed or failed.

    def is\_passed(self):

        pass\_mark = 50

        if self.average\_mark >= pass\_mark:

            return True

        else:

            return False

# Define the producer function.

def producer():

    # Generate a random student name.

    student\_name = random.choice(names)

    # Generate a random student ID.

    student\_id = random.randint(1000000000, 9999999999)

    # Generate a random programme.

    programme = random.choice(programmes)

    # Generate a list of courses.

    courses = []

    for i in range(random.randint(1, 5)):

        course = random.choice(courses\_list)

        courses.append(course)

    # Generate a list of marks for each course.

    marks = []

    for course in courses:

        m = random.randint(1, 100)

        marks.append(m)

    # Create an ITstudent object.

    student = ITstudent(student\_name, student\_id, programme, courses, marks)

    # Create an XML file for the student information.

    xml = f"""<?xml version="1.0" encoding="UTF-8"?>

    <student>

        <name>{student.student\_name}</name>

        <id>{student.student\_id}</id>

        <programme>{student.programme}</programme>

        <courses>

            {"".join([f"<course>{course}</course>" for course in student.courses])}

                     <marks>

                       {"".join([f"<mark>{mark}</mark>" for mark in student.marks])}

                     </marks>

        </courses>

    </student>"""

    # Generate a random number between 1-10 for the XML file name.

    xml\_file\_name = f"student{random.randint(1, 10)}.xml"

    # Write the XML file to the directory shared with the buffer.

    with open(xml\_file\_name, "w") as f:

        f.write(xml)

    # Insert the XML file name and the integer 1 into the buffer.

    buffer.append((xml\_file\_name, 1))

# Define the consumer function.

def consumer():

    # Check if the buffer is empty.

    if len(buffer) == 0:

        return

    # Get the first element from the buffer.

    xml\_file\_name, integer = buffer.pop(0)

    # Open the XML file and read the student information.

    with open(xml\_file\_name, "r") as f:

        xml = f.read()

    # Unwrap the XML file.

    student\_name = xml.split("<name>")[1].split("</name>")[0]

    student\_id = xml.split("<id>")[1].split("</id>")[0]

    programme = xml.split("<programme>")[1].split("</programme>")[0]

    courses = [[course.split(">")[1].split("<")[0]].split("<")[0] for course in xml.split("<course>")[1:]]

    marks = [int(mark.split(">")[1].split("<")[0]) for mark in xml.split("<mark>")[1:]]

    # Create an ITstudent object.

    student = ITstudent(student\_name, student\_id, programme, courses, marks)

    # Calculate the average mark.

    student.average\_mark = student.calculate\_average\_mark()

    # Determine whether the student passed or failed.

    student.is\_passed = student.is\_passed()

    # Print the student information.

    print(f"Name: {student.student\_name}")

    print(f"ID: {student.student\_id}")

    print(f"Programme: {student.programme}")

    print(f"Courses: {', '.join(student.courses)}")

    print(f"Marks: {', '.join([str(mark) for mark in student.marks])}")

    print(f"Average mark: {student.average\_mark}")

    print(f"Pass/Fail: {student.is\_passed}")

    # Clear the XML file.

    with open(xml\_file\_name, "w") as f:

        pass

    # Remove the integer from the buffer.

    buffer.pop(0)

# Define the buffer.

buffer = []

# Define the list of names.

names = ["John", "Jane", "David", "Emma", "Oliver", "Lucas", "Mia", "Noah", "Liam", "William"]

# Define the list of programmes.

programmes = ["Bachelor of Science in Computer Science", "Master of Science in Computer Science", "Doctor of Philosophy in Computer Science"]

# Define the list of courses.

courses\_list = ["Data Structures", "Algorithms", "Operating Systems", "Database Systems", "Computer Networks", "Artificial Intelligence"]

# Define the producer and consumer processes.

import multiprocessing

producer\_process = multiprocessing.Process(target=producer)

consumer\_process = multiprocessing.Process(target=consumer)

# Start the producer and consumer processes.

producer\_process.start()

consumer\_process.start()

# Wait for the producer and consumer processes to finish.

producer\_process.join()

consumer\_process.join()

consumer\_process.join()