#Week 5

#1)student coursee enrollment

# creating a dictiornary

enrollments = [

(1001, 'CS101', 'S1'),

(1001, 'MATH101', 'S1'),

(1002, 'CS101', 'S1'),

(1002, 'MATH102', 'S2'),

(1003, 'CS202', 'S2'),

(1003, 'PHY101', 'S1'),

(1004, 'CS202', 'S2'),

(1004, 'CHEM101', 'S1'),

(1005, 'BIO101', 'S1'),

(1005, 'HIST101', 'S1'),

(1006, 'BIO102', 'S2'),

(1006, 'ENGL101', 'S1'),

(1007, 'ECON101', 'S1'),

(1007, 'PSY101', 'S1'),

(1008, 'ECON102', 'S2'),

(1008, 'SOC101', 'S1'),

(1009, 'PSY102', 'S2'),

(1009, 'SOC102', 'S2'),

(1010, 'CS101', 'S1'),

(1010, 'MATH101', 'S1')

]

course\_enrollments = {}

# Loop through the dictionary to print out each students name and their enrolled courses

for student\_id, course\_code, \_ in enrollments:

if student\_id not in course\_enrollments:

course\_enrollments[student\_id] = []

course\_enrollments[student\_id].append(course\_code)

for student\_id, courses in course\_enrollments.items():

print(f"Student ID: {student\_id} = Enrolled Courses: {': '.join(courses)}")

#2)class schedule

courses = [

('CS101', 'Introduction to Computer Science', 'Computer Science', 'S1'),

('CS202', 'Data Structures and Algorithms', 'Computer Science', 'S2'),

('MATH101', 'Calculus I', 'Mathematics', 'S1'),

('MATH102', 'Calculus II', 'Mathematics', 'S2'),

('PHY101', 'General Physics I', 'Physics', 'S1'),

('PHY102', 'General Physics II', 'Physics', 'S2'),

('CHEM101', 'General Chemistry I', 'Chemistry', 'S1'),

('CHEM102', 'General Chemistry II', 'Chemistry', 'S2'),

('BIO101', 'Biology I', 'Biology', 'S1'),

('BIO102', 'Biology II', 'Biology', 'S2'),

('HIST101', 'American History I', 'History', 'S1'),

('HIST102', 'American History II', 'History', 'S2'),

('ENGL101', 'English Composition I', 'English', 'S1'),

('ENGL102', 'English Composition II', 'English', 'S2'),

('ECON101', 'Principles of Economics', 'Economics', 'S1'),

('ECON102', 'Intermediate Microeconomics', 'Economics', 'S2'),

('PSY101', 'Introduction to Psychology', 'Psychology', 'S1'),

('PSY102', 'Developmental Psychology', 'Psychology', 'S2'),

('SOC101', 'Introduction to Sociology', 'Sociology', 'S1'),

('SOC102', 'Social Problems', 'Sociology', 'S2')

]

departments = {}

for course\_id, course\_name, department, semester in courses:

if department not in departments:

departments[department] = []

departments[department].append((department, course\_id, course\_name))

for department, course\_list in departments.items():

for \_, course\_id, course\_name in course\_list:

print(f"Course Name: {course\_name}, Department: {department}")

#3)lecture asighnment

lecturer\_courses = {

"Dr. John Doe": ["Introduction to Computer Science"],

"Mr. Michael Johnson": ["General Physics II"],

"Asst. Prof. Olivia Taylor": ["General Chemistry I"],

"Dr. Emily Brown": ["Calculus II"],

"Prof. David Lee": ["General Physics I"],

"Miss. Sophia Carter": ["Biology I", "Biology II"],

"Dr. Oliver Hernandez": ["English Composition I"],

"Prof. Isabella Garcia": ["Introduction to Sociology"],

"Prof. Evelyn Russell": ["Principles of Economics", "Intermediate Microeconomics"],

"Dr. Lucas Sanchez": ["Introduction to Psychology", "Developmental Psychology"],

"Asst. Prof. Liam Lopez": ["Social Problems"]

}

# Loop through the dictionary and print the teachers name and the course they teach

for lecturer, courses in lecturer\_courses.items():

print(f"Lectur name: {lecturer}, Course name: {', '.join(courses)}")

#session 6

class\_list = []

while True:

student\_name = input("Enter the student's name (or type 'exit' to finish): ")

if student\_name.lower() == 'exit':

break

class\_list.append(student\_name)

print(f"You have added {student\_name} to your class.")

# Print the final class list

print("\nFinal class list:")

for student in class\_list:

print(student)

# Room data

rooms = [

{"RoomNumber": 101, "Capacity": 15, "FloorNumber": "Ground Floor", "Location": "Building A"},

{"RoomNumber": 102, "Capacity": 15, "FloorNumber": "Ground Floor", "Location": "Building A"},

{"RoomNumber": 103, "Capacity": 20, "FloorNumber": "Ground Floor", "Location": "Building A"},

{"RoomNumber": 104, "Capacity": 20, "FloorNumber": "Ground Floor", "Location": "Building A"},

{"RoomNumber": 105, "Capacity": 25, "FloorNumber": "Ground Floor", "Location": "Building A"},

{"RoomNumber": 106, "Capacity": 25, "FloorNumber": "Ground Floor", "Location": "Building A"},

{"RoomNumber": 107, "Capacity": 30, "FloorNumber": "Ground Floor", "Location": "Building A"},

{"RoomNumber": 108, "Capacity": 30, "FloorNumber": "Ground Floor", "Location": "Building A"},

{"RoomNumber": 109, "Capacity": 30, "FloorNumber": "Ground Floor", "Location": "Building A"},

{"RoomNumber": 110, "Capacity": 10, "FloorNumber": "Ground Floor", "Location": "Building A"},

{"RoomNumber": 201, "Capacity": 10, "FloorNumber": "1st Floor", "Location": "Building A"},

{"RoomNumber": 202, "Capacity": 10, "FloorNumber": "1st Floor", "Location": "Building A"},

{"RoomNumber": 203, "Capacity": 25, "FloorNumber": "1st Floor", "Location": "Building A"},

{"RoomNumber": 204, "Capacity": 25, "FloorNumber": "1st Floor", "Location": "Building A"},

{"RoomNumber": 205, "Capacity": 30, "FloorNumber": "1st Floor", "Location": "Building A"},

{"RoomNumber": 206, "Capacity": 40, "FloorNumber": "1st Floor", "Location": "Building A"},

{"RoomNumber": 207, "Capacity": 40, "FloorNumber": "1st Floor", "Location": "Building A"},

{"RoomNumber": 208, "Capacity": 40, "FloorNumber": "1st Floor", "Location": "Building A"},

]

# Class list management

class\_list = []

max\_students = 10

active = True

while active:

student\_name = input("Enter the student's name (or type 'exit', 'quit', or '0' to finish): ")

if student\_name.lower() in ['exit', 'quit', '0']:

active = False

print(f"\nTotal number of students entered: {len(class\_list)}")

print("Students:")

for student in class\_list:

print(student)

continue

if len(class\_list) >= max\_students:

print(f"Maximum number of students ({max\_students}) reached. Exiting...")

break

class\_list.append(student\_name)

print(f"You have added {student\_name} to your class.")

# Room capacity check

while True:

try:

min\_capacity = int(input("Enter the minimum number of seats required for the class (or type '0' to exit): "))

if min\_capacity == 0:

break

suitable\_rooms = [room for room in rooms if room["Capacity"] >= min\_capacity]

if suitable\_rooms:

for room in suitable\_rooms:

print(f"Room {room['RoomNumber']} with capacity {room['Capacity']} is available on {room['FloorNumber']}, {room['Location']}.")

else:

print("No suitable rooms found for the requested capacity.")

except ValueError:

print("Please enter a valid number.")

# Infinite loop for additional input

print("\nPress CTRL-C to stop entering inputs.")

line\_count = 0

try:

while True:

u\_input = input("Enter a line of text: ")

print(u\_input)

line\_count += 1

except KeyboardInterrupt:

print(f"\nNo of lines entered : {line\_count}")