**Ministerul Educaţiei și Cercetării al Republicii Moldova**

**Colegiul Universitatii Tehnice a Moldovei**

# RAPORT

Lucrarea de laborator

*Asistenta pentru OOP*

A efectuat: C.Maxim

A verificat: Cătălin Coșeru

Chişinău - 2024

**Lucrare de laborator**

**Build a program loop, an interactive command line for the TUM Board to be able to do the next operations: • Faculty operations: 1. Create and assign a student to a faculty. 2. Graduate a student from a faculty. 3. Display current enrolled students (Graduates would be ignored). 4. Display graduates (Currently enrolled students would be ignored). 5. Tell or not if a student belongs to this faculty. • General operations: 1. Create a new faculty. 2. Search what faculty a student belongs to by a unique identifier (for example by email or a unique ID). 3. Display University faculties. 4. Display all faculties belonging to a field. (Ex. FOOD\_TECHNOLOGY)**

class University:

def \_\_init\_\_(self):

self.faculties = {}

def create\_faculty(self, faculty\_name, field):

self.faculties[faculty\_name] = {'field': field, 'students': []}

def assign\_student\_to\_faculty(self, faculty\_name, student\_id, student\_name):

if faculty\_name in self.faculties:

self.faculties[faculty\_name]['students'].append({'id': student\_id, 'name': student\_name})

print(f"Student {student\_name} assigned to {faculty\_name}")

else:

print(f"Faculty {faculty\_name} does not exist.")

def graduate\_student(self, student\_id):

for faculty in self.faculties.values():

students = faculty['students']

for student in students:

if student['id'] == student\_id:

students.remove(student)

print(f"Student {student['name']} graduated.")

return

print(f"Student with ID {student\_id} not found.")

def display\_enrolled\_students(self, faculty\_name):

if faculty\_name in self.faculties:

students = self.faculties[faculty\_name]['students']

print(f"Enrolled students in {faculty\_name}:")

for student in students:

print(f"ID: {student['id']}, Name: {student['name']}")

else:

print(f"Faculty {faculty\_name} does not exist.")

def display\_graduates(self):

print("List of Graduates:")

for faculty in self.faculties.values():

for student in faculty['students']:

print(f"ID: {student['id']}, Name: {student['name']}")

def check\_student\_belongs\_to\_faculty(self, student\_id, faculty\_name):

if faculty\_name in self.faculties:

students = self.faculties[faculty\_name]['students']

for student in students:

if student['id'] == student\_id:

print(f"Yes, student {student['name']} belongs to {faculty\_name}.")

return

print(f"No, student with ID {student\_id} does not belong to {faculty\_name}.")

else:

print(f"Faculty {faculty\_name} does not exist.")

def search\_faculty\_by\_student\_id(self, student\_id):

for faculty\_name, faculty\_info in self.faculties.items():

for student in faculty\_info['students']:

if student['id'] == student\_id:

print(f"Student with ID {student\_id} belongs to {faculty\_name} faculty.")

return

print(f"Student with ID {student\_id} not found in any faculty.")

def display\_university\_faculties(self):

print("List of University Faculties:")

for faculty\_name, faculty\_info in self.faculties.items():

print(f"Faculty: {faculty\_name}, Field: {faculty\_info['field']}")

def display\_faculties\_by\_field(self, field):

print(f"Faculties belonging to {field} field:")

for faculty\_name, faculty\_info in self.faculties.items():

if faculty\_info['field'] == field:

print(f"Faculty: {faculty\_name}, Field: {faculty\_info['field']}")

def main():

university = University()

while True:

print("\n===== TUM BOARD COMMAND LINE =====")

print("Faculty Operations:")

print("1. Create and assign a student to a faculty.")

print("2. Graduate a student from a faculty.")

print("3. Display current enrolled students.")

print("4. Display graduates.")

print("5. Check if a student belongs to a faculty.")

print("\nGeneral Operations:")

print("6. Create a new faculty.")

print("7. Search what faculty a student belongs to by a unique identifier.")

print("8. Display University faculties.")

print("9. Display all faculties belonging to a field.")

print("0. Exit")

choice = input("Enter your choice (0-9): ")

if choice == '1':

faculty\_name = input("Enter faculty name: ")

student\_id = input("Enter student ID: ")

student\_name = input("Enter student name: ")

university.assign\_student\_to\_faculty(faculty\_name, student\_id, student\_name)

elif choice == '2':

student\_id = input("Enter student ID to graduate: ")

university.graduate\_student(student\_id)

elif choice == '3':

faculty\_name = input("Enter faculty name: ")

university.display\_enrolled\_students(faculty\_name)

elif choice == '4':

university.display\_graduates()

elif choice == '5':

faculty\_name = input("Enter faculty name: ")

student\_id = input("Enter student ID: ")

university.check\_student\_belongs\_to\_faculty(student\_id, faculty\_name)

elif choice == '6':

faculty\_name = input("Enter new faculty name: ")

field = input("Enter field of the new faculty: ")

university.create\_faculty(faculty\_name, field)

print(f"Faculty {faculty\_name} created.")

elif choice == '7':

student\_id = input("Enter student ID: ")

university.search\_faculty\_by\_student\_id(student\_id)

elif choice == '8':

university.display\_university\_faculties()

elif choice == '9':

field = input("Enter field to display faculties: ")

university.display\_faculties\_by\_field(field)

elif choice == '0':

print("Exiting TUM BOARD. Goodbye!")

break

else:

print("Invalid choice. Please enter a number between 0 and 9.")

if \_\_name\_\_ == "\_\_main\_\_":

main()