**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**

**As one can notice each file has a different extension .txt, .png and .py. The user is informed of the files changed status since the last snapshot time. From this output one can gather: test.txt has experienced no change since the snapshot time of 2023-10-11, 09:01:28, while image.png and python\_script.py did change since the snapshot. 2 Task: VERY IMPORTANT! 1. No third party libraries are allowed, you can use only the ones available in your programming language. 2. You are free to create as many classes as you need to achieve a well structured working system. 3. Limitation of concepts: You are required to structure you’re program using the concepts of Inheritance and Polymorphism (Runtime and Compile time Polymorphism is a must) 4. USING GIT: You have to commit each significant change in your program. Failing doing so results in a 2 point deduction, plus writing a 2 page report on "Git usefulness". If there’s no report provided, laboratory isn’t accepted.import os**

**import time**

class File:

def \_\_init\_\_(self, name, extension, last\_modified):

self.name = name

self.extension = extension

self.last\_modified = last\_modified

def has\_changed(self, last\_snapshot\_time):

return self.last\_modified > last\_snapshot\_time

def \_\_str\_\_(self):

return f"{self.name}.{self.extension}"

class TextFile(File):

def \_\_init\_\_(self, name, last\_modified):

super().\_\_init\_\_(name, "txt", last\_modified)

class ImageFile(File):

def \_\_init\_\_(self, name, last\_modified):

super().\_\_init\_\_(name, "png", last\_modified)

class PythonFile(File):

def \_\_init\_\_(self, name, last\_modified):

super().\_\_init\_\_(name, "py", last\_modified)

class FolderSnapshot:

def \_\_init\_\_(self, folder\_path):

self.folder\_path = folder\_path

self.snapshot\_time = time.time()

self.files = self.scan\_folder()

def scan\_folder(self):

files = []

for filename in os.listdir(self.folder\_path):

file\_path = os.path.join(self.folder\_path, filename)

if os.path.isfile(file\_path):

last\_modified = os.path.getmtime(file\_path)

if filename.endswith(".txt"):

files.append(TextFile(filename[:-4], last\_modified))

elif filename.endswith(".png"):

files.append(ImageFile(filename[:-4], last\_modified))

elif filename.endswith(".py"):

files.append(PythonFile(filename[:-3], last\_modified))

return files

def detect\_changes(self):

print("Files changed status since the last snapshot time:")

for file in self.files:

if file.has\_changed(self.snapshot\_time):

print(f"{file} has changed since the snapshot.")

else:

print(f"{file} has experienced no change since the snapshot time of {time.ctime(self.snapshot\_time)}.")

folder\_path = "your\_folder\_path\_here"

snapshot = FolderSnapshot(folder\_path)

snapshot.detect\_changes()