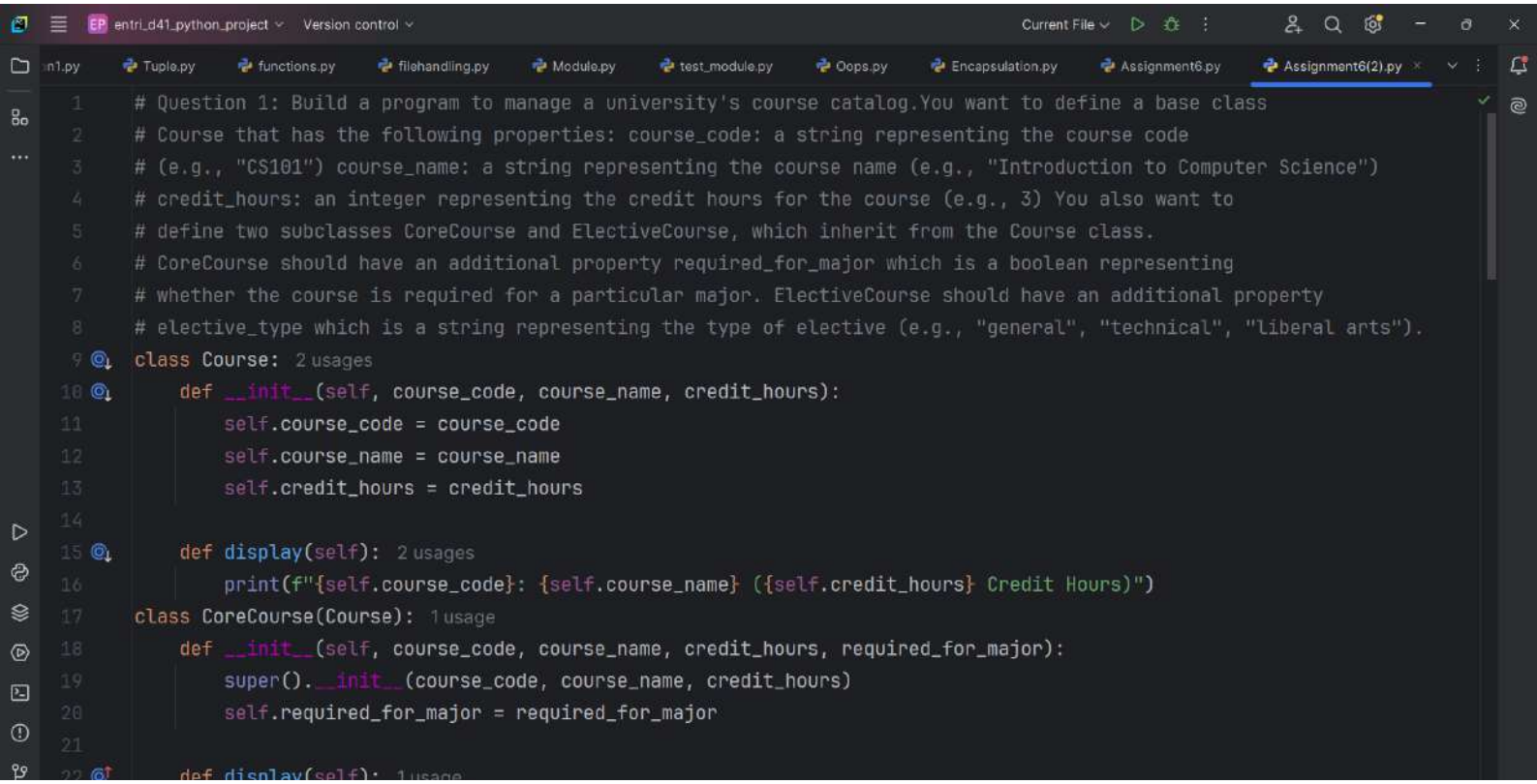
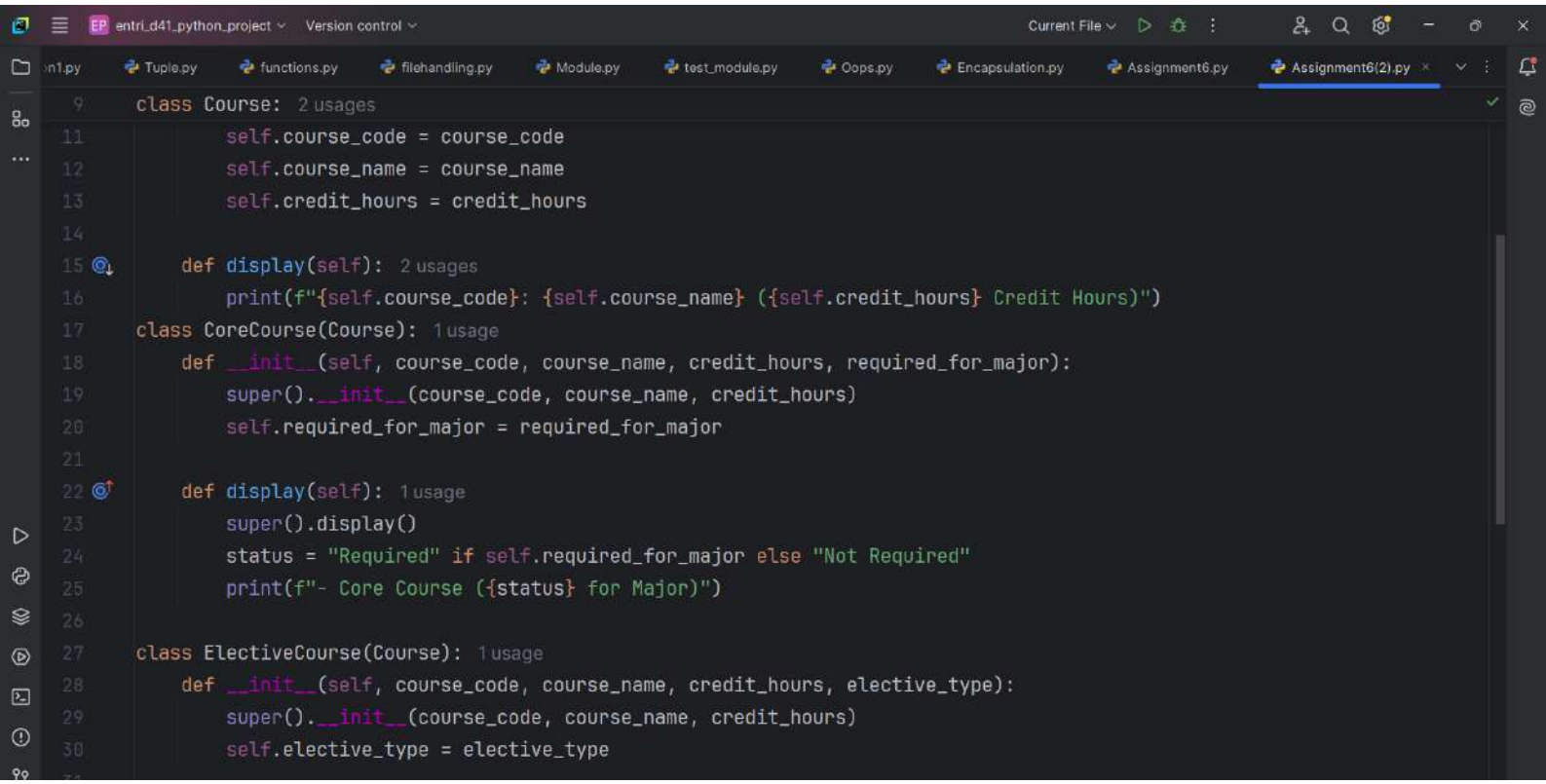
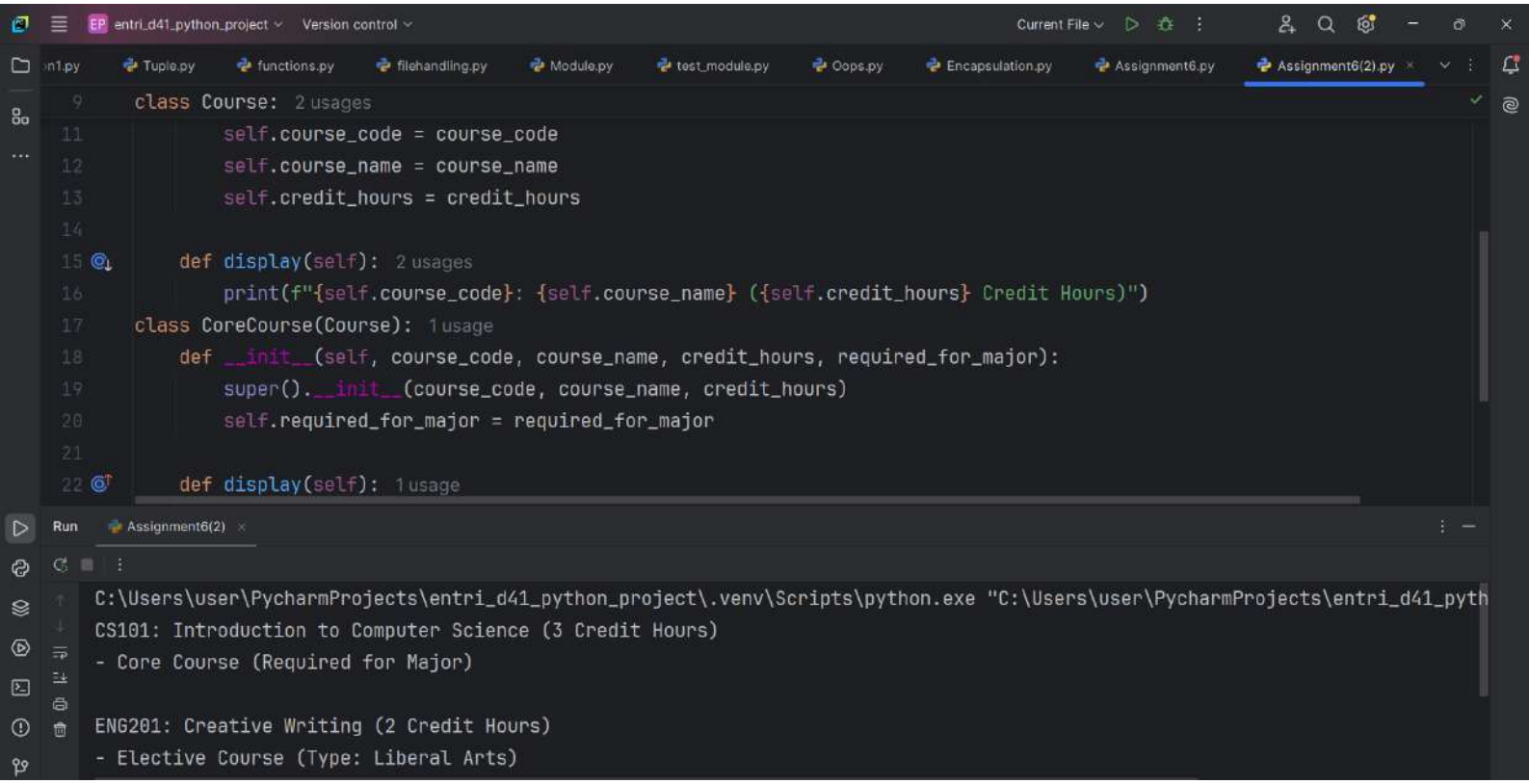
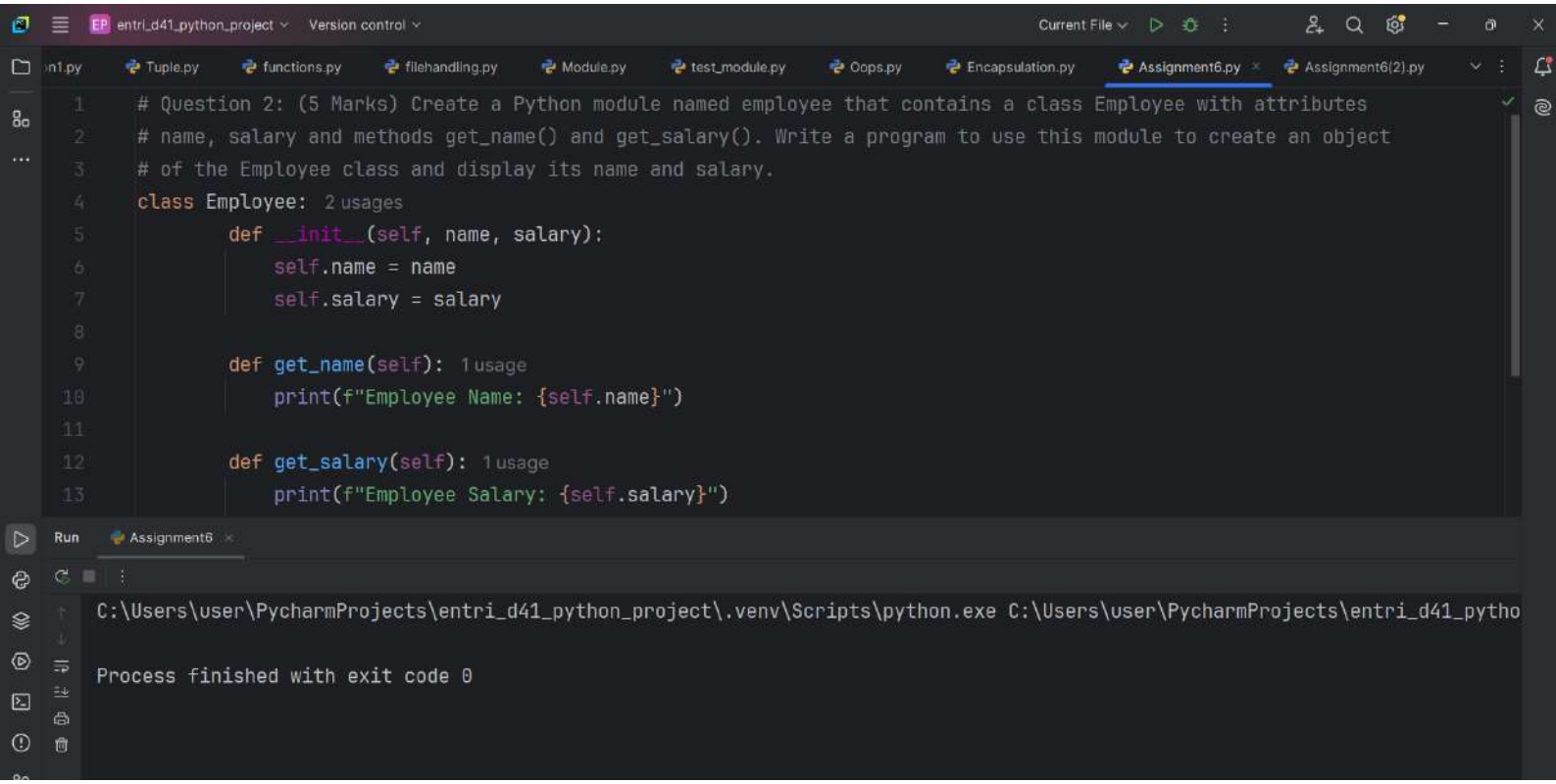

entri_d41_python_project
Version control
Current File
n1.py
Tuple.py
functions.py
filehandling.py
Module.py
test_module.py
Oops.py
Encapsulation.py
Assignment6.py
Assignment6(2).py
Question 1: Build a program to manage a university's course catalog.You want to define a base class
Course that has the following properties: course_code: a string representing the course code
(e.g., "CS101") course_name: a string representing the course name (e.g., "Introduction to Computer Science")
credit_hours: an integer representing the credit hours for the course (e.g., 3) You also want to
define two subclasses CoreCourse and ElectiveCourse, which inherit from the Course class.
CoreCourse should have an additional property required_for_major which is a boolean representing
whether the course is required for a particular major. ElectiveCourse should have an additional property
elective_type which is a string representing the type of elective (e.g., "general", "technical", "liberal arts").
class Course: 2 usages
def __init__(self, course_code, course_name, credit_hours):
self.course_code = course_code
self.course_name = course_name
self.credit_hours = credit_hours
def display(self): 2 usages
print(f"{self.course_code}: {self.course_name} ({self.credit_hours} Credit Hours)")
class CoreCourse(Course): 1 usage
def __init__(self, course_code, course_name, credit_hours, required_for_major):
super().__init__(course_code, course_name, credit_hours)
self.required_for_major = required_for_major
def display(self): 1 usage


entri_d41_python_project
Version control
Current File
Tuple.py
functions.py
filehandling.py
Module.py
test_module.py
Oops.py
Encapsulation.py
Assignment6.py
Assignment6(2).py
class Course: 2 usages
self.course_code = course_code
self.course_name = course_name
self.credit_hours = credit_hours
def display(self): 2 usages
print(f"{self.course_code}: {self.course_name} ({self.credit_hours} Credit Hours)")
class CoreCourse(Course): 1 usage
def __init__(self, course_code, course_name, credit_hours, required_for_major):
super().__init__(course_code, course_name, credit_hours)
self.required_for_major = required_for_major
def display(self): 1 usage
super().display()
status = "Required" if self.required_for_major else "Not Required"
print(f"- Core Course ({status} for Major)")
class ElectiveCourse(Course): 1 usage
def __init__(self, course_code, course_name, credit_hours, elective_type):
super().__init__(course_code, course_name, credit_hours)
self.elective_type = elective_type


entri_d41_python_project
Version control
Current File
Tuple.py
functions.py
filehandling.py
Module.py
test_module.py
Oops.py
Encapsulation.py
Assignment6.py
Assignment6(2).py
class Course: 2 usages
self.course_code = course_code
self.course_name = course_name
self.credit_hours = credit_hours
def display(self): 2 usages
print(f"{self.course_code}: {self.course_name} ({self.credit_hours} Credit Hours)")
class CoreCourse(Course): 1 usage
def __init__(self, course_code, course_name, credit_hours, required_for_major):
super().__init__(course_code, course_name, credit_hours)
self.required_for_major = required_for_major
def display(self): 1 usage
Run
Assignment6(2)
C:\Users\user\PycharmProjects\entri_d41_python_project\.venv\Scripts\python.exe "C:\Users\user\PycharmProjects\entri_d41_pyth
CS101: Introduction to Computer Science (3 Credit Hours)
- Core Course (Required for Major)
ENG201: Creative Writing (2 Credit Hours)
- Elective Course (Type: Liberal Arts)


entri_d41_python_project
Version control
Current File
Tuple.py
functions.py
filehandling.py
Module.py
test_module.py
Oops.py
Encapsulation.py
Assignment6.py
Assignment6(2).py
Question 2: (5 Marks) Create a Python module named employee that contains a class Employee with attributes
name, salary and methods get_name() and get_salary(). Write a program to use this module to create an object
of the Employee class and display its name and salary.
class Employee: 2 usages
def __init__(self, name, salary):
self.name = name
self.salary = salary
def get_name(self): 1 usage
print(f"Employee Name: {self.name}")
def get_salary(self): 1 usage
print(f"Employee Salary: {self.salary}")
Run
Assignment6
C:\Users\user\PycharmProjects\entri_d41_python_project\.venv\Scripts\python.exe C:\Users\user\PycharmProjects\entri_d41_pytho
Process finished with exit code 0

```python
from Assignment6 import Employee
emp = Employee( name: "Alice",  salary: 50000)
emp.get_name()
emp.get_salary()
```

Run Assignment6(2)

```
Employee Name: Alice
Employee Salary: 50000

Process finished with exit code 0
```