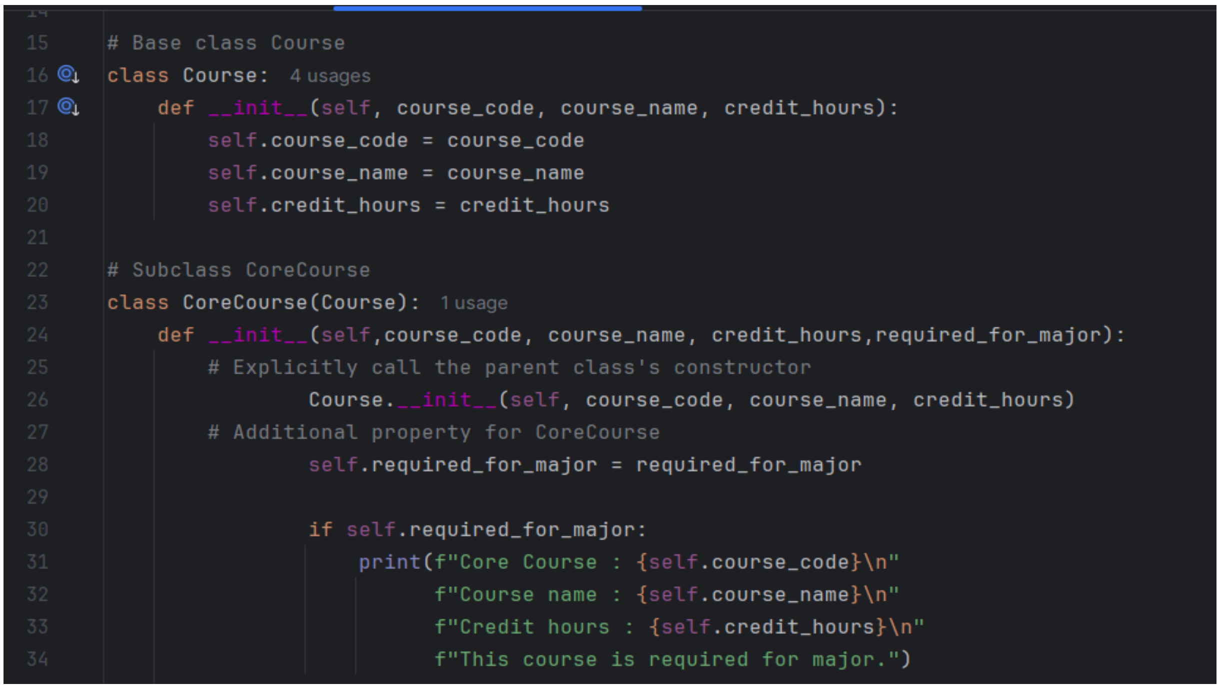
Modules and Classes

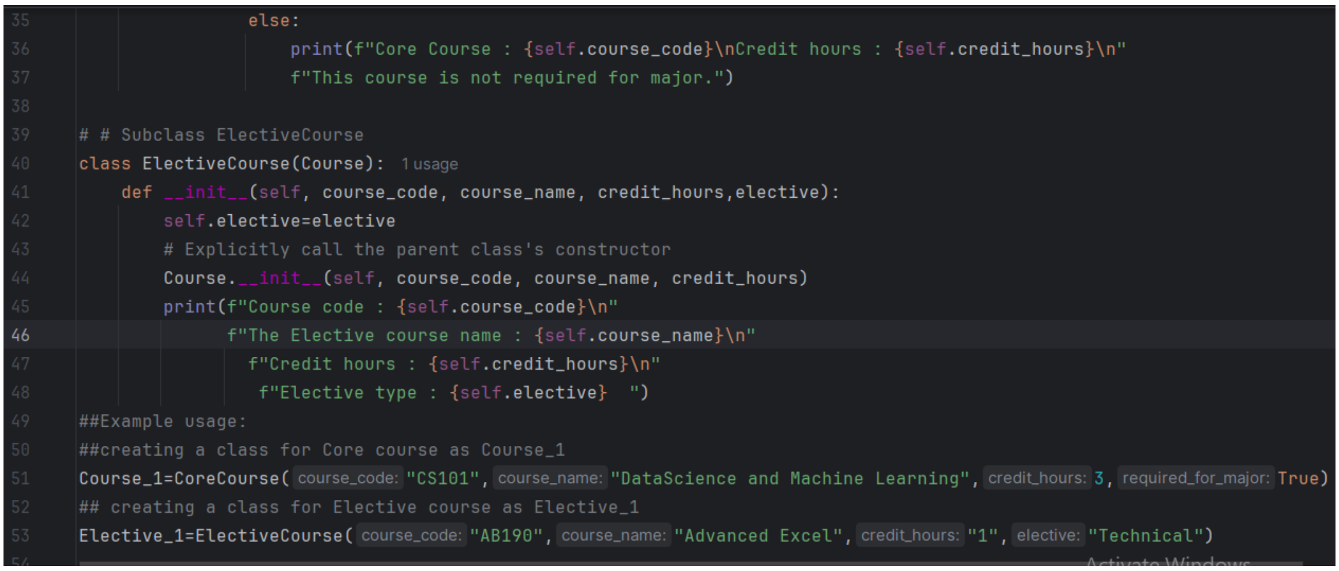
Python Assignment

```
1. """Question 1: (5 Marks)
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").

"""
```

Code :

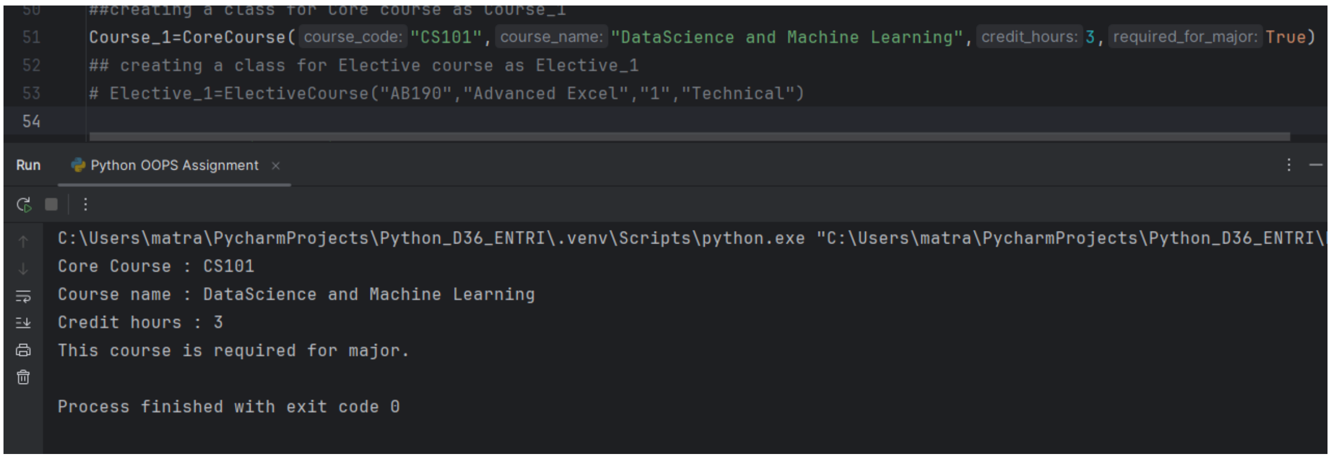


```python
# Base class Course
class Course:  4 usages
    def __init__(self, course_code, course_name, credit_hours):
        self.course_code = course_code
        self.course_name = course_name
        self.credit_hours = credit_hours

# Subclass CoreCourse
class CoreCourse(Course):  1 usage
    def __init__(self,course_code, course_name, credit_hours,required_for_major):
        # Explicitly call the parent class's constructor
                Course.__init__(self, course_code, course_name, credit_hours)
        # Additional property for CoreCourse
                self.required_for_major = required_for_major

                if self.required_for_major:
                    print(f"Core Course : {self.course_code}\n"
                          f"Course name : {self.course_name}\n"
                          f"Credit hours : {self.credit_hours}\n"
                          f"This course is required for major.")
```



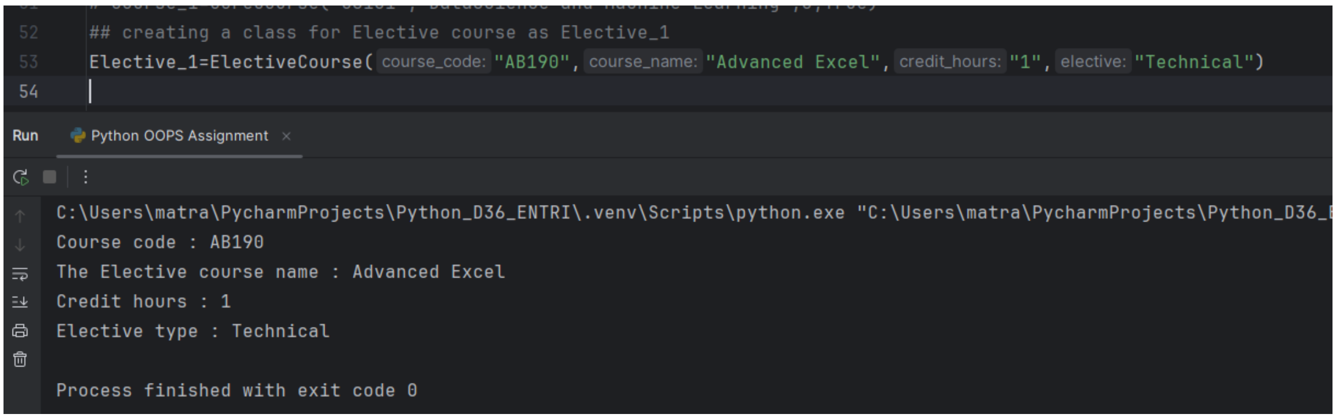
```python
                else:
                    print(f"Core Course : {self.course_code}\nCredit hours : {self.credit_hours}\n"
                    f"This course is not required for major.")

# # Subclass ElectiveCourse
class ElectiveCourse(Course):  1 usage
    def __init__(self, course_code, course_name, credit_hours,elective):
        self.elective=elective
        # Explicitly call the parent class's constructor
        Course.__init__(self, course_code, course_name, credit_hours)
        print(f"Course code : {self.course_code}\n"
              f"The Elective course name : {self.course_name}\n"
                f"Credit hours : {self.credit_hours}\n"
                 f"Elective type : {self.elective}  ")
##Example usage:
##creating a class for Core course as Course_1
Course_1=CoreCourse( course_code: "CS101", course_name: "DataScience and Machine Learning", credit_hours: 3, required_for_major: True)
## creating a class for Elective course as Elective_1
Elective_1=ElectiveCourse( course_code: "AB190", course_name: "Advanced Excel", credit_hours: "1", elective: "Technical")
```

OutPut:



```
##creating a class for Core course as Course_1
Course_1=CoreCourse( course_code: "CS101", course_name: "DataScience and Machine Learning", credit_hours: 3, required_for_major: True)
## creating a class for Elective course as Elective_1
# Elective_1=ElectiveCourse("AB190","Advanced Excel","1","Technical")
```

```
C:\Users\matra\PycharmProjects\Python_D36_ENTRI\.venv\Scripts\python.exe "C:\Users\matra\PycharmProjects\Python_D36_ENTRI\
Core Course : CS101
Course name : DataScience and Machine Learning
Credit hours : 3
This course is required for major.

Process finished with exit code 0
```

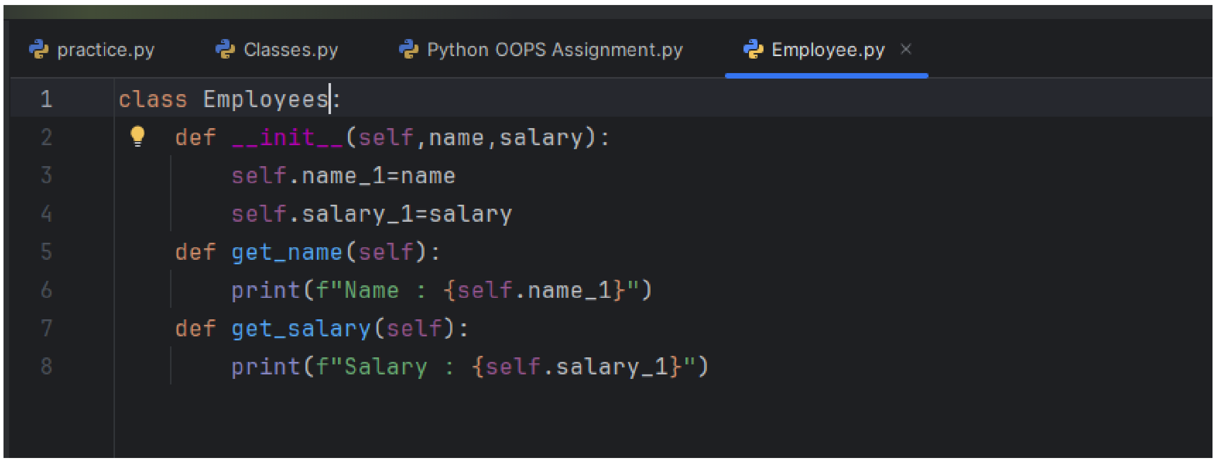


```
## creating a class for Elective course as Elective_1
Elective_1=ElectiveCourse( course_code: "AB190", course_name: "Advanced Excel", credit_hours: "1", elective: "Technical")
```

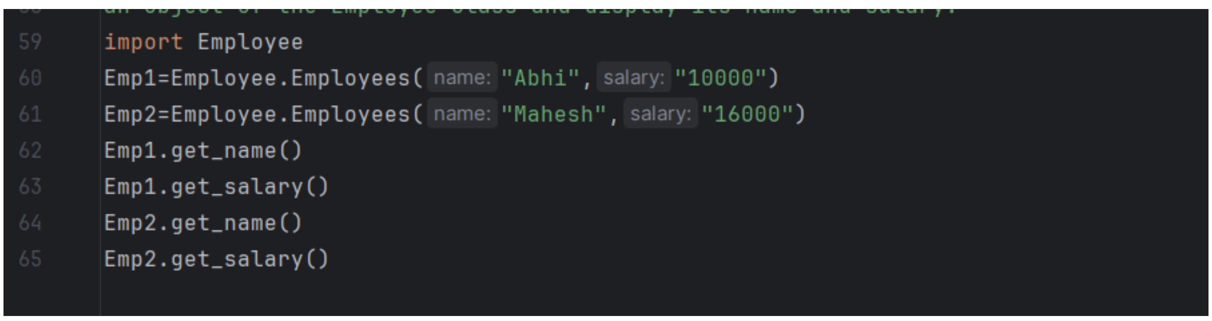
```
C:\Users\matra\PycharmProjects\Python_D36_ENTRI\.venv\Scripts\python.exe "C:\Users\matra\PycharmProjects\Python_D36_
Course code : AB190
The Elective course name : Advanced Excel
Credit hours : 1
Elective type : Technical

Process finished with exit code 0
```

```
2. """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."""
```

Code :

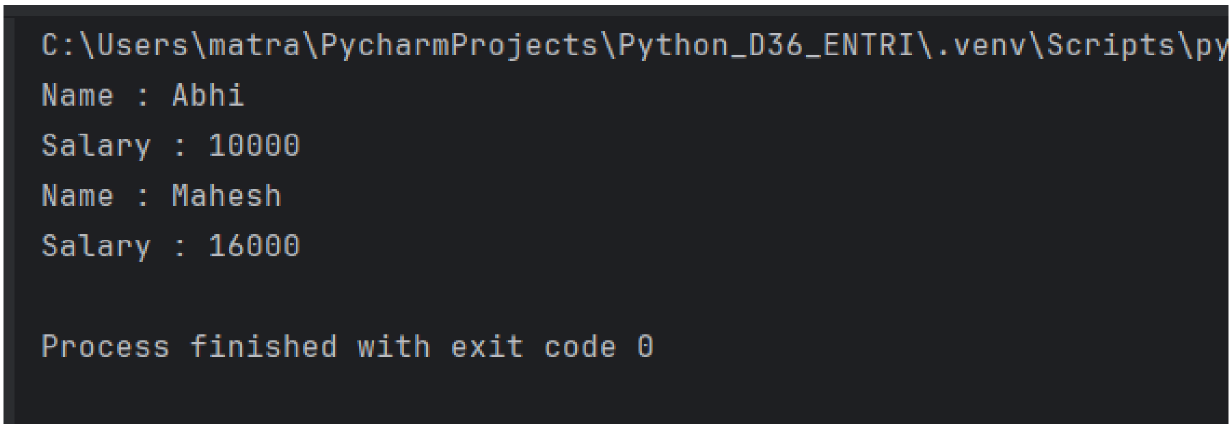


```python
class Employees:
    def __init__(self,name,salary):
        self.name_1=name
        self.salary_1=salary
    def get_name(self):
        print(f"Name : {self.name_1}")
    def get_salary(self):
        print(f"Salary : {self.salary_1}")
```



```python
an object of the Employee class and display its name and salary.
import Employee
Emp1=Employee.Employees( name: "Abhi", salary: "10000")
Emp2=Employee.Employees( name: "Mahesh", salary: "16000")
Emp1.get_name()
Emp1.get_salary()
Emp2.get_name()
Emp2.get_salary()
```

OutPut :



```
C:\Users\matra\PycharmProjects\Python_D36_ENTRI\.venv\Scripts\py
Name : Abhi
Salary : 10000
Name : Mahesh
Salary : 16000

Process finished with exit code 0
```