TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

PULCHOWK CAMPUS

A REPORT ON

OOP in Python Language

SUBMITTED BY:

PRASTAV PANDEY (081BEL059)

SUBMITTED TO:

DEPARTMENT OF ELECTRONICS & COMPUTER ENGINEERING

QUESTION 1:

Create a class Book with attributes title, author, and price. Write a constructor to initialize these values and create an object with sample data.

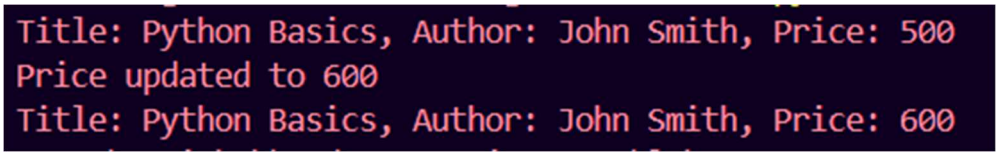
● Add a method display_info() to the Book class that prints the book's title, author, and price. Call this method using a Book object.

● Add a method update_price(new_price) to the Book class that updates the book's price. Demonstrate how to use it with an object.

CODE 1:

```
class Book:
    def __init__(self, title, author, price):
        self.title = title
        self.author = author
        self.price = price

    def display_info(self):
        print(f"Title: {self.title}, Author: {self.author}, Price: {self.price}")

    def update_price(self, new_price):
        self.price = new_price
        print(f"Price updated to {self.price}")



b1 = Book("Python Basics", "John Smith", 500)
b1.display_info()
b1.update_price(600)
b1.display_info()
```

OUTPUT 1:



OUESTION 2:

Create a class Student with attributes name and marks. Create three objects of the class and display their details using a method show_details().

CODE 2:

```
class Student:
    def __init__(self, name, marks):
        self.name = name
        self.marks = marks

    def show_details(self):
        print(f"Name: {self.name}, Marks: {self.marks}")



s1 = Student("Alice", 85)
s2 = Student("Bob", 90)
s3 = Student("Charlie", 78)

s1.show_details()
s2.show_details()
s3.show_details()
```

OUTPUT 2:

```
Name: Alice, Marks: 85
Name: Bob, Marks: 90
Name: Charlie, Marks: 78
```

QUESTION 3:

Create a class BankAccount with attributes account_holder, account_number, and balance.

- Add methods deposit(amount) and withdraw(amount) that update the balance.
- Add a method show_balance() that prints the current balance.
- Create an object and perform a deposit, a withdrawal, and show the balance.

CODE 3:

```
class BankAccount:
    def __init__(self, account_holder, account_number, balance=0):
        self.account_holder = account_holder
        self.account_number = account_number
        self.balance = balance

    def deposit(self, amount):
        self.balance += amount
        print(f"Deposited {amount}")

    def withdraw(self, amount):
        if amount <= self.balance:
            self.balance -= amount
            print(f"Withdrew {amount}")
        else:
            print("Insufficient balance")
```

```
    def show_balance(self):
        print(f"Current balance: {self.balance}")



acc1 = BankAccount("David", "12345", 1000)
acc1.show_balance()
acc1.deposit(500)
acc1.withdraw(300)
acc1.show_balance()
```

OUTPUT 3:

```
Current balance: 1000
Deposited 500
Withdrew 300
Current balance: 1200
```