Task 1 Vehicle

```
Көлік таңдаңыз (car/bike): car
Көлік жолда жүреді.
Көлік бензин қолданады.
```

```
Көлік таңдаңыз (car/bike): bike
Велосипед педальмен жүреді.
Велосипед адамның энергиясын қолданады.
```

1 Абстрактты класс және әдістер

```
class Vehicle(ABC):
    @abstractmethod
    def move(self): pass
    @abstractmethod
    def fuel(self): pass
```

- `Vehicle` – тек басқа кластарға негіз болады.
- `move` және `fuel` – міндетті түрде әр мұрагер класс орындауы керек әдістер.

---

2 Мұрагер класстарда әдісті жүзеге асыру

```
class Car(Vehicle):
    def move(self):
        print("Көлік жолда жүреді.")
    def fuel(self):
        print("Көлік бензин қолданады.")
```

- Мұнда абстрактты әдістер нақты іске асырылған.

```
class Bicycle(Vehicle):
    def move(self):
        print("Велосипед педальмен жүреді.")
    def fuel(self):
        print("Велосипед адамның энергиясын қолданады.")
```

- Сол сияқты, `Bicycle` класында да жүзеге асырылады.

---

3 Пайдаланушы таңдауы мен шақыру

```
choice = input("Көлік таңдаңыз (car/bike): ")

if choice == "car":
    v = Car()
elif choice == "bike":
    v = Bicycle()
else:
    print("Қате таңдау!")
    exit()

v.move()
v.fuel()
```

- Пайдаланушы не таңдаса, сол объект жасалып, оның `move()` және `fuel()` әдістері шақырылады.

Task 2 db

1 Абстрактты класс `DB`

```
class DB(ABC):
    @abstractmethod
    def connect(self): pass

    @abstractmethod
    def query(self): pass

    @abstractmethod
    def close(self): pass
```

- `DB` – **абстрактты класс**, тек шаблон ретінде жұмыс істейді.
- `connect`, `query`, `close` – міндетті түрде әр мұрагер класс орындауы керек әдістер.

---

2 Мұрагер класстар

```
class SQLite(DB):
    def connect(self):
        print("SQLite: қосылды.")
    def query(self):
        print("SQLite: сұраныс орындалды.")
    def close(self):
        print("SQLite: жабылды.")
```

- `SQLite` – `DB`-ны мұрагер етеді.
- Әр әдіс нақты іске асырылған: қосылу, сұраныс орындау, жабылу.

```
class Postgre(DB):
    def connect(self):
        print("PostgreSQL: қосылды.")
    def query(self):
        print("PostgreSQL: сұраныс орындалды.")
    def close(self):
        print("PostgreSQL: жабылды.")
```

- Сол сияқты `Postgre` класы үшін де нақты әдістер бар.

---

3 Пайдаланушы таңдауы және шақыру

```
db_choice = input("Дерекқорды таңдаңыз (sqlite/pg): ")

db = SQLite() if db_choice == "sqlite" else Postgre()

db.connect()
db.query()
db.close()
```

- `input` арқылы пайдаланушы дерекқорды таңдайды.
- `SQLite` немесе `Postgre` объектісі жасалады.
- Содан соң `connect()`, `query()`, `close()` әдістері шақырылады.

```
Дерекқорды таңдаңыз (sqlite/pg): sqlite
SQLite: қосылды.
SQLite: сұраныс орындалды.
SQLite: жабылды.
```

```
Дерекқорды таңдаңыз (sqlite/pg): pg
PostgreSQL: қосылды.
PostgreSQL: сұраныс орындалды.
PostgreSQL: жабылды.
```

Task 3 log

Абстрактты класс `Logger`

- `start()` – лог басталғанын көрсетеді.
- `log(msg)` – міндетті абстрактты әдіс, әр мұрагерде жүзеге асады.

2 Мұрагер класстар

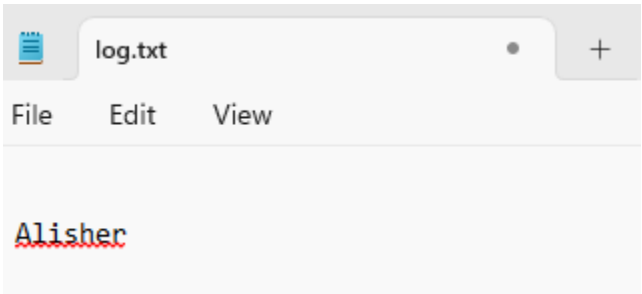
- `FileLogger` → хабарды файлға жазып, хабарлайды.
- `ConsoleLogger` → хабарды тек консольге шығарады.

3 Пайдаланушы таңдауы

```
logger = FileLogger() if log_type == "file" else ConsoleLogger()
logger.start()
logger.log(msg)
```

- Пайдаланушы таңдайды, сәйкес объект жасалады, лог басталады және хабар жазылады.

```
Логгер түрін таңдаңыз (file/console): file
Хабар жазыңыз: Alisher
=== ЛОГ БАСТАЛДЫ ===
Файлға жазылды.
```




```
Логгер түрін таңдаңыз (file/console): console
Хабар жазыңыз: Alisher
=== ЛОГ БАСТАЛДЫ ===
Консоль: Alisher
```

Task 4 polif

1 Абстрактты класс `Vehicle`

```
class Vehicle(ABC):
    @abstractmethod
    def move(self): pass
```

- `Vehicle` – шаблон класс.
- `move()` – міндетті әдіс, әр мұрагер орындауы керек.

---

2 Мұрагер кластар

```
class Car(Vehicle):
    def move(self):
        print("Көлік жүреді.")
```

- `Car`, `Bicycle`, `Airplane`, `Train` – әрқайсысы `Vehicle`-ді мұрагер етеді.
- `move()` әдісі нақты әрекетті көрсетеді (жүреді, ұшады).

---

3 Пайдаланушы таңдауы

```
choice = input("Көлік таңдаңыз: ")

classes = {
    "car": Car,
    "bike": Bicycle,
    "airplane": Airplane,
    "train": Train
}

if choice not in classes:
    print("Қате таңдау!")
    exit()

obj = classes[choice]()
obj.move()
```

- `input` арқылы пайдаланушы таңдайды.
- Сөздік (`classes`) арқылы сәйкес класс шақырылады.
- Объект жасалып, `move()` әдісі орындалады.

```
Көліктер тізімі: car, bike, airplane, train
Көлік таңдаңыз: car
Көлік жүреді.
```

```
Көліктер тізімі: car, bike, airplane, train
Көлік таңдаңыз: bike
Велосипед жүреді.
```

```
Көліктер тізімі: car, bike, airplane, train
Көлік таңдаңыз: airplane
Ұшақ ұшады.
```

```
Көліктер тізімі: car, bike, airplane, train
Көлік таңдаңыз: train
Пойыз жүріп келеді.
```

Task 5 Oplata

```
Төлем жүйесін таңдаңыз (paypal/kaspi): paypal
PayPal: авторизация ОК.
PayPal: төлем орындалды.
PayPal: ақша қайтарылды.
```

```
Төлем жүйесін таңдаңыз (paypal/kaspi): kaspi
Kaspi: расталды.
Kaspi: төлем өтті.
Kaspi: қайтарылды.
```

1 Абстрактты класс `Payment`

- `authorize()` – төлемді рұқсаттау.
- `process()` – төлемді орындау.
- `refund()` – ақшаны қайтару.
- Бұл әдістер **әр мұрагерде міндетті түрде жүзеге асады**.

---

2 Мұрагер кластар

- `PayPal` → PayPal төлем жүйесіне арналған нақты әрекеттер.
- `Kaspi` → Kaspi төлем жүйесіне арналған нақты әрекеттер.
- Әр класс абстрактты әдістерді нақты іске асырады.

3️⃣ Пайдаланушы таңдауы мен шақыру

```
payment_type = input("Төлем жүйесін таңдаңыз (paypal/kaspi): ")

pay = PayPal() if payment_type == "paypal" else Kaspi()

pay.authorize()
pay.process()
pay.refund()
```

- `input` арқылы пайдаланушы төлем жүйесін таңдайды.
- Таңдаған жүйеге сәйкес объект жасалады.
- Әдістерді шақырып, төлем процесін имитациялайды.