به نام ایزد یکتا

برنامه سازی پیشرفته - 4041

استاد درس : آقای دکترعلی جاویدانی

اعضای تیم دستیاران دانشجویی : مهدیس یعقوب انصاری - کیانا جابری - مبینا صفری متین امیرپناه فر- شهاب ثمری شمس - محمد متین سلیمانی - پارسا دولت آبادی - نیما مخملی

سلام بچه‌ها! 👋
به مینی‌پروژهٔ چهارم خوش اومدین 🙂

در این تمرین قراره وارد دنیایی بشیم که:

- هر **وسیلهٔ نقلیه** یک موجودیت مستقل است
- هر **قانون ترافیکی** یک تصمیم منطقی است
- و طراحی درست، جایگزین کدهای پر از `if / else` می‌شود

در این مینی‌پروژه، تمرکز **بر مدل‌سازی قوانین و تعامل اشیاء** است،
نه شبیه‌سازی گرافیکی یا فیزیکی حرکت خودروها.

هدف کلی تمرین

هدف این تمرین، سنجش توانایی شما در:

- تحلیل مسئله و استخراج موجودیت‌های درست
- طراحی روابط صحیح بین کلاس‌ها
- استفادهٔ معنادار از **Polymorphism**
- استفادهٔ منطقی از **Operator Overloading**
- درک کاربرد واقعی **static**
- پیاده‌سازی یک سیستم شیءگرا با مسئولیت‌های مشخص

نکتهٔ مهم:
هدف این تمرین «درست طراحی شدن» است، نه فقط «کار کردن برنامه».

مسئلهٔ تمرین

فرض کنید می‌خواهیم هستهٔ ساده‌ای از یک سیستم مدیریت ترافیک شهری را طراحی کنیم؛
سیستمی که در آن:

- وسایل نقلیه مختلف رفتار متفاوتی دارند
- قوانین ترافیکی قابل توسعه هستند
- تصمیم حرکت خودروها وابسته به قوانین محیط است، نه شرط‌های ثابت

قابلیت‌های مورد انتظار سیستم

1 وسایل نقلیه (Vehicle)

هر وسیلهٔ نقلیه دارای:

- شناسه
- پلاک
- سرعت فعلی

نکتهٔ مهم:

- همهٔ وسایل نقلیه یکسان رفتار نمی‌کنند
- تصمیم حرکت باید بر اساس قوانین محیط باشد

2 عناصر کنترلی ترافیک (TrafficElement)

عناصر کنترلی مشخص می‌کنند:

- آیا یک وسیلهٔ نقلیه اجازهٔ حرکت دارد یا خیر

ویژگی‌ها:

- قوانین باید **چندریخت** باشند
- افزودن قانون جدید نباید نیاز به تغییر کدهای قبلی داشته باشد

3 تقاطع (Intersection)

هر تقاطع شامل:

- مجموعه‌ای از وسایل نقلیه
- مجموعه‌ای از قوانین ترافیکی

وظایف:

- بررسی امکان حرکت هر وسیله
- اعمال قوانین
- تشخیص وضعیت‌هایی مانند ترافیک یا توقف کامل
- چاپ وضعیت تقاطع

کلاس‌های الزامی

کلاس Vehicle (abstract)

ویژگی‌ها:

```
int id;

string plate;

int speed;
```

متدهای الزامی:

```
virtual bool canMove(const TrafficElement&) const = 0;

virtual void move();

virtual void printStatus() const;
```

زیرکلاس‌های الزامی:

- `Car`
- `Bus`
- `EmergencyVehicle`

مثال:
وسیلهٔ امدادی می‌تواند از چراغ قرمز عبور کند، اما خودروی معمولی نه.

🔹 کلاس TrafficElement (abstract)

متدهای الزامی:

```
virtual bool allows(const Vehicle&) const = 0;

virtual void printRule() const;
```

زیرکلاس‌های پیشنهادی:

- `TrafficLight`
- `StopSign`
- `SpeedLimitZone`

🔹 Operator Overloading (الزامی)

استفاده از عملگرها باید **منطقی و معنادار** باشد:

- `Vehicle + int` → افزایش سرعت با رعایت محدودیت‌ها
- `Vehicle < Vehicle` → مقایسهٔ اولویت عبور
- `TrafficLight == LightState` → بررسی وضعیت چراغ

استفادهٔ غیرمنطقی از operator نمرهٔ منفی دارد.

🔹 کلاس Intersection

ویژگی‌ها:

```
vector<Vehicle*> vehicles;

vector<TrafficElement*> rules;
```

وظایف:

- بررسی قوانین
- تصمیم‌گیری دربارهٔ حرکت وسایل نقلیه
- چاپ وضعیت تقاطع

🔹 استفادهٔ الزامی از static

- شمارندهٔ کل وسایل نقلیه
- (در صورت نیاز) متغیرهای سراسری مرتبط با شبیه‌سازی

نکات مهم طراحی

- ارسال اشیاء به توابع با `reference` یا `const reference`
- استفادهٔ صحیح از `virtual` و `override`
- عدم استفاده از شرط‌های زنجیره‌ای به‌جای چندریختی
- عدم hard-code شدن قوانین
- تمرکز بر **مدل ذهنی سیستم**، نه تعداد کلاس‌ها

ساختار پیشنهادی پروژه

```
HW4-StudentNumber/
├── include/
│   ├── Vehicle.h
│   ├── Car.h
│   ├── Bus.h
│   ├── EmergencyVehicle.h
│   ├── TrafficElement.h
│   ├── TrafficLight.h
│   ├── StopSign.h
│   ├── SpeedLimitZone.h
│   ├── Intersection.h
│
├── src/
│   ├── Vehicle.cpp
│   ├── Car.cpp
│   ├── Bus.cpp
│   ├── EmergencyVehicle.cpp
│   ├── TrafficElement.cpp
│   ├── TrafficLight.cpp
│   ├── StopSign.cpp
│   ├── SpeedLimitZone.cpp
│   ├── Intersection.cpp
│   └── main.cpp
```

مثال سادهٔ سناریو (فقط برای درک مسئله)

- یک `EmergencyVehicle` در تقاطعی با چراغ قرمز و محدودیت سرعت قرار دارد
- سیستم باید تشخیص دهد:
  - آیا مجاز به حرکت است؟
  - سرعت نهایی آن چقدر است؟

(پیاده‌سازی سناریو آزاد است)

معیارهای ارزیابی (Rubric پیشنهادی)

- طراحی کلاس‌ها و روابط: **30**٪
- استفادهٔ صحیح از چند ریختی : **30**٪
- Operator Overloading منطقی: **15**٪
- استفادهٔ درست از استاتیک : **15**٪
- خوانایی و ساختار پروژه: **10**٪

📥 نحوه تحویل

- تمامی فایل‌های مرتبط با تمرین، شامل **سورس‌کد و هرگونه مستندات یا توضیحات** تکمیلی، باید در یک پوشه با نام زیر قرار گیرند:
  `HW4-StudentNumber`
- این پوشه را فشرده کرده و با فرمت `.zip` ارسال نمایید.
- **ارسال فایل تنها از طریق بخش "تمرین چهارم" در کلاس درس، در سامانه کوئرا صورت می‌گیرد.**

لطفاً پیش از اتمام مهلت مقرر، از آپلود صحیح فایل اطمینان حاصل فرمایید.