# برنامه‌سازی پیشرفته

**تمرین کامپیوتری شماره 2**

مدرس: رامتین خسروی

طراحان: پریسا یحیی‌پور، امیرحسین عارف‌زاده، طاها مجلسی، امیررضا نادی، مهراد لیویان، حسام رمضانیان

مهلت تحویل: چهارشنبه ۲ آبان ۱۴۰۳، ساعت ۲۳:۵۹

## مقدمه

هدف از این تمرین آشنایی شما با طراحی بالا به پایین[1] یک مسئله است. با توجه به حجم پروژه لازم است که قبل از شروع پیاده‌سازی زمانی را به طراحی اختصاص دهید. در غیر این صورت در هنگام پیاده‌سازی با مشکل مواجه می‌شوید. بنابراین ابتدا به چگونگی شکستن این مسئله به مسائل کوچک‌تر و پخش کردن مسئولیت‌ها میان قسمت‌های مختلف برنامه فکر کنید.

برای آشنایی بیشتر شما با این نوع طراحی می‌توانید به ویدیویی که در بخش محتوای دستیاران آموزشی در صفحه درس یا وب‌سایت درس مراجعه کنید.

## برنامه مدیریت پارکینگ

در این برنامه، شما وظیفه دارید یک سیستم مدیریت پارکینگ پیاده‌سازی کنید. این سیستم با ورود هر خودرو، بر اساس ویژگی‌های آن، مکان‌های پارک مناسب را به کاربر پیشنهاد می‌دهد و کاربر می‌تواند از میان آن‌ها یک مکان را انتخاب کند. همچنین، هنگام خروج خودرو، برنامه هزینه پارکینگ را با توجه به ویژگی‌های خودرو و مدت زمانی که در پارکینگ حضور داشته است، محاسبه می‌کند.

[1] Top-Down Design

# قالب فایل‌های ورودی

اطلاعات مربوط به ماشین‌ها، محل پارک و جدول قیمت در سه فایل جداگانه قرار دارند که باید اطلاعات موجود در آن‌ها را در ابتدای اجرا برنامه خوانده و ذخیره کنید. نوع فایل‌های ورودی به صورت CSV[2] هستند و برای آشنایی با این نوع فایل‌ها می‌توانید این لینک را بررسی کنید.

**تذکر: تضمین می‌شود داده‌های تمام فایل‌های CSV مطابق توضیحات می‌باشند و مقدار دیگری وجود ندارد.**

در ادامه شرح هر یک از فایل‌ها آمده است.

## فایل ماشین‌ها

در این فایل دو ستون name و size وجود دارد. اندازه ماشین‌ها به سه صورت مشخص می‌شود:

- 1: ماشین‌های کوچک
- 2: ماشین‌های متوسط
- 3: ماشین‌های بزرگ

ماشین‌ها لزوما به ترتیب اندازه در فایل قرار ندارند.

| نوع داده | توضیحات | نام ویژگی |
|---|---|---|
| رشته | نام ماشین | name |
| عدد طبیعی | اندازه ماشین | size |

نمونه فایل محل پارک‌ها

```
name,size
Toyota_Corolla,2
Honda_Fit,1
Ford_Expedition,3
```

[2] Comma-Separated Values

## فایل محل پارک‌ها

در این فایل سه ستون type، size، id وجود دارد. نوع محل پارک به سه صورت مشخص می‌شود:

- normal: این محل پارک معمولی است و ویژگی خاصی ندارد
- covered: این محل پارک سقف دارد که ماشین را از باران و برف محافظت می‌کند
- CCTV: این محل پارک مجهز به دوربین مدار بسته می‌باشد

اندازه محل پارک مانند فایل ماشین‌ها مشخص می‌شود.

شناسه محل پارک‌ها لزوما به ترتیب و صعودی نیست. در ابتدای برنامه تمام محل‌های پارک خالی هستند.

| نام ویژگی | توضیحات | نوع داده |
|---|---|---|
| id | شناسه محل پارک | عدد طبیعی |
| size | اندازه محل پارک | عدد طبیعی |
| type | نوع محل پارک | رشته |

نمونه فایل محل پارک‌ها

```
id,size,type
101,1,normal
301,2,covered
201,3,CCTV
```

## فایل قیمت‌ها

در این فایل سه ستون price_per_day، static_price، size وجود دارد. این داده‌ها مربوط به محل پارک normal هستند.

اندازه محل پارک مشابه اندازه در فایل ماشین‌ها می‌باشد.

| نام ویژگی | توضیحات | نوع داده |
|---|---|---|
| size | اندازه محل پارک | عدد طبیعی |

| | | |
|---|---|---|
| عدد طبیعی | هزینه ثابت | static_price |
| عدد طبیعی | هزینه به ازای هر ساعت | price_per_day |

نمونه فایل قیمت‌ها

```
size,static_price,price_per_day
1,250,50
2,300,70
3,350,80
```

# نحوه اجرای برنامه

در زمان اجرای برنامه مسیر فایل‌های ورودی به ترتیب (ابتدا مسیر فایل ماشین‌ها، سپس مسیر فایل محل پارک‌ها و سپس مسیر فایل قیمت‌ها) از طریق آرگومان خط فرمان به برنامه داده می‌شود. برای آشنایی با این نوع آرگومان‌ها، می‌توانید این لینک را مشاهده کنید.

نمونه ورودی

```
./Parking.out </path/to/cars/file> </path/to/parking/file>
</path/to/price/file>
```

# دستورات برنامه

## دستور درخواست محل پارک

در این دستور، کاربر از طریق خط فرمان، با وارد کردن نام ماشین، لیستی از محل‌های پارک مناسب با اندازه ماشین وارد شده را مشاهده می‌کند که در حال حاضر خالی هستند.
تضمین می‌شود نام ماشین وارد شده توسط کاربر، در فایل CSV ماشین‌ها وجود دارد و به همان شکل در دستور وارد می‌شود.

قالب دستور

```
request_spot <Car’s name>
```

پس از وارد کردن این دستور، اطلاعات تمام محل‌های پارک مناسب بر اساس شناسه به صورت صعودی نمایش داده می‌شوند. تضمین می‌شود که در زمان اجرای این دستور حداقل یک محل پارک مناسب در برنامه برای ماشین مورد نظر وجود خواهد داشت.
توجه داشته باشید که هزینه هر محل پارک بر اساس ویژگی‌های محل پارک نمایش داده میشود که در جلوتر نحوه محاسبه هزینه هر محل پارک براساس ویژگی‌های آن گفته خواهد شد.

قالب خروجی

```
<ID>: <type> <static_price> <price_per_hour>
…
```

نمونه ورودی

```
request_spot Honda_Fit
```

نمونه خروجی

```
101: normal 250 50
```

## دستور رزرو محل پارک

پس از مشاهده لیست محل‌های پارک موجود، کاربر با وارد کردن این دستور و شناسه مورد نظر، محل پارک را رزرو می‌کند، دقت کنید که ممکن است این دستور بدون وارد کردن دستور درخواست محل پارک وارد شود و الزامی نیست همواره پشت سر هم وارد شوند.

قالب دستور

```
assign_spot <Spot’s ID>
```

پس از وارد کردن دستور، پیامی به نشانه موفق بودن عملیات نمایش داده می‌شود. تضمین می‌شود شناسه وارد شده جزو محل‌های قابل پارک کردن می‌باشد و خطایی اینجا رخ نمی‌دهد.

| قالب خروجی |
| --- |
| `Spot <Spot's ID> is occupied now.` |

| نمونه ورودی |
| --- |
| `assign_spot 101` |

| نمونه خروجی |
| --- |
| `Spot 101 is occupied now.` |

## دستور خروج ماشین

با استفاده از این دستور و وارد کردن شناسه محل پارک، کاربر ماشین پارک شده در این محل را خارج می‌کند.

| قالب دستور |
| --- |
| `checkout <Spot's ID>` |

پس از وارد کردن این دستور، پیام خالی شدن محل پارک به همراه هزینه استفاده از محل پارک محاسبه شده و به کاربر نمایش داده می‌شود.

| قالب خروجی |
| --- |
| `Spot <Spot's ID> is free now.`<br>`Total cost: <cost>` |

## نحوه محاسبه هزینه

برای استفاده از هر محل پارک یک هزینه ثابت در نظر گرفته شده که در فایل مربوط به محل پارک در ستون static_price قرار دارد. کاربر بدون توجه به تعداد روزهایی که در پارکینگ بوده، باید این هزینه را پرداخت کند. هزینه اضافی کاربر بر اساس تعداد روزهایی که از پارکینگ استفاده کرده محاسبه می‌شود.

هزینه = هزینه ثابت + (تعداد روزها × هزینه به ازای هر روز)

هزینه مربوط به محل پارک normal در فایل قیمت‌ها قرار دارد. در صورتی که نوع محل پارک غیر از normal باشد، هزینه ثابت و هزینه به ازای هر روز به شکل زیر محاسبه می‌شود:

- covered:
  - هزینه ثابت = 50 + هزینه ثابت normal
  - هزینه به ازای هر روز = 30 + هزینه به ازای هر روز normal
- CCTV:
  - هزینه ثابت = 80 + هزینه ثابت normal
  - هزینه به ازای هر روز = 60 + هزینه به ازای هر روز normal

| نمونه ورودی 1 |
| --- |
| `checkout 101` |

فرض کنیم کاربر 5 روز از پارکینگ استفاده کرده است. در این صورت خروجی به شکل زیر خواهد بود.

| نمونه خروجی 1 |
| --- |
| `Spot 101 is free now.`<br>`Total cost: 500` |

توضیح هزینه: محل پارک با شناسه 101 طبق فایل ورودی برای اندازه ماشین کوچک (1) در نظر گرفته شده و نوع آن normal است. مطابق فایل قیمت‌ها، محل پارک با این ویژگی ها دارای هزینه ثابت (static_price) به مقدار 250 و هزینه به ازای هر روز (price_per_day) به مقدار 50 می‌باشد. بنابراین هزینه کل به شکل زیر خواهد بود:

هزینه = 250 + 5 × 50 = 500

نمونه ورودی 2

```
checkout 301
```

فرض کنیم کاربر 5 روز از پارکینگ استفاده کرده است. در این صورت خروجی به شکل زیر خواهد بود.

نمونه خروجی 2

```
Spot 301 is free now.
Total cost: 850
```

توضیح هزینه: محل پارک با شناسه 301 طبق فایل ورودی برای اندازه ماشین متوسط (2) در نظر گرفته شده و نوع آن covered است. مطابق فایل قیمت‌ها، محل پارک normal دارای هزینه ثابت (static_price) به مقدار 300 و هزینه به ازای هر روز (price_per_day) به مقدار 70 می‌باشد. بنابراین برای محاسبه هزینه‌های نوع covered خواهیم داشت:

- هزینه ثابت = 300 + 50 = 350
- هزینه به ازای هر روز = 70 + 30 = 100

بنابراین هزینه کل به شکل زیر خواهد بود:

هزینه = 350 + 5 × 100 = 850

## جلو بردن زمان

در این دستور زمان به تعداد روز های وارد شده می‌گذرد. توجه کنید در ابتدای برنامه مقدار اولیه روز صفر است. مقدار وارد شده برای روز حتما طبیعی است.

قالب دستور

```
pass_time <Number of days>
```

نمونه ورودی

```
pass_time 3
```

## مثال

نمونه ورودی

```
request_spot Ford_Expedition
assign_spot 201
pass_time 3
assign_spot 101
pass_time 4
checkout 201
checkout 101
```

نمونه خروجی

```
201: CCTV 430 140
Spot 201 is occupied now.
Spot 101 is occupied now.
Spot 201 is free now.
Total cost: 1410
Spot 101 is free now.
Total cost: 450
```

# نکات و نحوه تحویل

- برای تحویل این پروژه، لازم است کد خود را در یک مخزن[3] در GitHub بارگذاری کنید و سپس لینک مخزن به همراه Hash آخرین کامیت[4] را در صفحه eLearn درس بارگذاری نمایید.
  - مراحل ثبت‌نام در GitHub:
    - 1. از طریق لینک ارائه‌شده به صفحه ثبت‌نام وارد شوید.
    - 2. پس از وارد کردن ایمیل، رمز عبور، و نام کاربری، یک ایمیل حاوی کد احراز هویت برای شما ارسال می‌شود.
    - 3. این کد را در صفحه ثبت‌نام وارد کنید و سپس با نام کاربری و رمز عبور خود وارد سایت شوید.
  - ساخت مخزن جدید:
    - 1. از طریق منوی بالای صفحه، گزینه "New repository" را انتخاب کنید.
    - 2. در صفحه باز شده، نام را به AP-F03-A2-<SID> تغییر دهید (دقت کنید که به جای SID از شماره دانشجویی خود استفاده کنید) و گزینه "Private" را انتخاب کنید. همچنین، کاربر **@AP-UT** را به مخزن اضافه کنید.
  - بارگذاری فایل‌ها:
    - 1. پس از ایجاد مخزن، روی "uploading an existing file" کلیک کنید.
    - 2. کد خود را به‌صورت یک فایل با نام A2-<SID>.cpp در این صفحه بارگذاری کنید. دقت کنید که این فایل باید در صفحه اول مخزن باشد و نباید در هیچ پوشه‌ای قرار داده شده باشد.
  - نهایی‌سازی و ارسال:
    - 1. پس از بارگذاری فایل، در صفحه مخزن، گزینه "commit" را انتخاب کنید.
    - 2. در صفحه جدید، با استفاده از آیکون کپی، SHA آخرین commit را کپی کنید.
    - 3. لینک مخزن و SHA کپی‌شده را در بخش مربوطه در صفحه eLearn درس وارد کنید.

نمونه متن خواسته شده در سامانه ای‌لرن (سه بخش <username> و <repository_name> و <last_commit_hash> را جایگزین کنید):

---

[3] Repository

[4] Commit

https://github.com/<username>/<repository_name>
<last_commit_hash>

- برنامه شما باید در سیستم عامل لینوکس و با مترجم g++ با استاندارد c++20 ترجمه و در زمان معقول برای ورودی های آزمون اجرا شود.
- **بخش مهمی از ارزیابی برنامه شما به درستی عملکرد آن می‌پردازد. بنابراین به اندازه کافی ورودی‌های آزمایشی طراحی کنید تا درستی خروجی در حالت‌های مختلف آزموده شود.**
- درستی برنامه شما از طریق آزمون‌های خودکار سنجیده می‌شود. به این ترتیب، لازم است خروجی تولید شده از نظر بزرگی و کوچکی حروف، رعایت فاصله‌ها، عدم وجود خروجی‌های اضافه، ... دقیقا مانند نمونه‌های داده شده باشد. بنابراین پیشنهاد می‌شود که با استفاده از ابزارهایی مانند diff فرمت خروجی برنامه خود را با خروجی‌هایی که در اختیارتان قرار داده شده است مطابقت دهید.
- هدف این تمرین یادگیری شماست. لطفا تمرین را خودتان انجام دهید. در صورت کشف تقلب مطابق سیاست درس با آن برخورد خواهد شد.
- توجه کنید که رعایت نکردن ساختار گفته شده در نام‌گذاری مخزن، فایل کد و آپلود موارد خواسته شده باعث کسر 5 درصد از نمره شما خواهد شد.

# نمرات

- تمیزی کد
  - رعایت کردن نام‌گذاری صحیح و انسجام[5]
  - عدم وجود کد تکراری
  - رعایت دندانه‌گذاری[6]
  - عدم استفاده از متغیرهای گلوبال
  - استفاده **صحیح** از متغیرهای ثابت به جای Magic Value-ها
- درستی کد
  - آزمون‌های خودکار
- طراحی
  - شکستن مناسب و مرحله به مرحله مسئله
  - ذخیره اطلاعات در ساختار داده‌های مناسب
  - ساختاردهی کد در قالب توابع کوتاه که فقط یک کار را انجام می‌دهند

*دقت کنید که موارد ذکر شده لزوما کل نمره شما را تشکیل نمی‌دهند و ممکن است با تغییراتی همراه باشند.*

[5] Consistency

[6] Indentation