برنامه‌سازی پیشرفته

تمرین کامپیوتری شماره ۶

مدرس: رامتین خسروی

طراحان: امیررضا نادی، مجید صادقی نژاد، سهیل حاجیان‌منش، عرفان میرشمس

مهلت تحویل: یکشنبه ۶ خرداد ۱۴۰۳، ساعت ۲۳:۵۹

مقدمه

این پروژه به جمع‌بندی آموخته‌های شما در این درس می‌پردازد. انتظار می‌رود مهارت‌هایی را که در تمرین‌های پیشین و سایر بخش‌های درس آموخته‌اید، در پیاده‌سازی این پروژه به کار گیرید.

در این پروژه شما باید یک سامانه رزرو رستوران را پیاده‌سازی کنید. قابلیت رزرو غذا و میز، مشاهده اطلاعات رستوران ها و ... از معدود امکاناتی است که در طول این فازها پیاده‌سازی خواهید کرد.

نکته قابل توجه در این پروژه این است که بهتر است پروژه به صورت incremental (تدریجی) پیاده‌سازی و آزموده شود. به طوری که ابتدا یک ساختار کلی از پروژه پیاده‌سازی شود و سپس دستورات مختلف مرحله به مرحله به آن اضافه گردد.

شرح تمرین

در این تمرین، شما باید سامانه‌ای را برای رزرو رستوران پیاده‌سازی کنید. این سامانه به کاربران امکان می‌دهد تا رستوران‌های مختلف و منوی آن‌ها را مشاهده کنند. کاربران می‌توانند در صورت وجود ظرفیت، رستوران مورد نظر خود را انتخاب کرده و با تعیین تعداد نفرات و زمان دلخواه، رزرو خود را ثبت کنند. همچنین امکان سفارش غذا نیز در این سامانه فراهم شده است.

در ابتدای اجرای برنامه، شما باید یک سری اطلاعات را از روی مجموعه داده[1]هایی که در قالب CSV[2] به شما داده می‌شود بخوانید و در برنامه خود ذخیره کنید. در ادامه، توضیح هر کدام از این مجموعه داده‌ها آمده است.

همچنین دقت داشته باشید که سامانه ما رزرو و همه دستورات را برای همان روز انجام می‌دهد و نیاز نیست پیچیدگی‌های ساعت و روز را در نظر بگیرید.

قالب فایل‌های ورودی

اطلاعات رستوران‌ها و محله‌ها در فایل‌های جدا قرار داده می‌شود. مسیر این فایل‌ها به ترتیب در قالب آرگومان‌های خط فرمان به برنامه داده می‌شود. برای آشنایی بیشتر با این آرگومان‌ها، می‌توانید به این لینک مراجعه کنید. دقت کنید که در هر دو فایل تضمین می‌شود نام رستوران و محله‌ها یکتا است.

پیکربندی نحوه اجرای برنامه

```
./UTaste </path/to/restaurants/file.csv>
</path/to/districts/file.csv>
```

فایل رستوران‌ها

این فایل شامل پنج ستون است که نشان‌دهنده نام رستوران، محله‌ای که رستوران در آن واقع است، لیست غذاهایی که در منوی رستوران است همراه با قیمت آن‌ها، ساعت شروع و پایان کار رستوران و میزهای موجود در رستوران می‌باشد. ساعت شروع به کار و ساعت پایان کار رستوران یک عدد صحیح بین 1 تا 24 می باشد و حتما ساعت پایان کار پس از ساعت شروع کار است (زمان پایان کار حتما قبل 24 است و به روز بعد نمی‌رود). مقادیر غذاها که از چند بخش تشکیل شده با جداکننده‌ای[3] از هم جدا شدند که در پیکربندی و مثال مشخص شده‌اند (توجه کنید که براکت‌ها صرفا برای خوانایی است و در فایل اصلی همان‌گونه که از

---

[1] Dataset

[2] Comma-Separated Values

[3] Delimiter

نمونه‌ها مشخص است وجود ندارد). با ساخت میزها به هر یک، یک شناسه مشخص تعلق می‌گیرد که از یک شروع می‌شود و تا تعداد میزها ادامه دارد.

پیکربندی فایل رستوران‌ها

```
name,district,foods:prices,opening_time,closing_time,number_of_tables
name1,district1,[food1:price1;food2:price2;...],opening_time1,closing_time1,number_of_tables1
…
```

نمونه فایل رستوران‌ها

```
name,district,foods:prices,opening_time,closing_time,number_of_tables
sib,Shariati,pitza:220;berger:190;hotdog:185,9,23,45
maman joon,Gotham,pitza:290;berger:180;sandwich:95;food with space in name:20,11,1,20
```

فایل محله‌ها

در هر خط این فایل همسایه‌های هر محله مشخص شده است. در ابتدای هر خط اسم محله می‌آید و در جلوی آن به ترتیب همسایه‌ها بر اساس نزدیکی نوشته شده‌اند؛ اولین همسایه نزدیک‌ترین و آخرین همسایه دورترین است. همچنین نام محله ها می‌تواند دارای فاصله باشد.

پیکربندی فایل محله‌ها

```
district,neighbors
district1,neighbor1;neighbor2;…;neighbor{n-1}
district2,neighbor1;neighbor2;…;neighbor{n-1}
…
```

نمونه فایل محله‌ها

```
district,neighbors
Azadi,Behboodi;SatarKhan;Shadman
Saadat Abad,Punak;Shahrak Gharb;Tajrish
Enghelab,Jomhouri;Karegar
```

انواع دستورات

در این فاز، منطق برنامه در قالب تعدادی دستور که در ادامه توضیح داده شده است پیاده‌سازی می‌شود. روند استفاده از برنامه به این شکل است که کاربر در برنامه شما با استفاده از رابط خط فرمان[4]، دستوری همراه با آرگومان‌های لازم برای اجرای آن در ورودی استاندارد وارد می‌کند. به عنوان مثال، برای گرفتن یک لیست از اطلاعات موجودیت‌ها از دستور GET و در صورت گرفتن یک عنصر خاص از موجودیت مورد نظر بعد از دستور، شناسه[5] موجودیتی که مایل به گرفتن اطلاعات آن است را نیز وارد می‌کند تا دستور مورد نظرش اجرا شود.

همینطور در نظر داشته باشید که این اطلاعات، یعنی آرگومان‌های هر دستور، پس از علامت ? در دستور می‌آیند و نیز ترتیب خاصی برای آن‌ها وجود ندارد؛ به این معنا که لزومی ندارد آرگومان‌ها به همان ترتیبی که در توضیحات هر دستور گفته شده، وارد شوند. توجه کنید که برای دستوراتی که آرگومان نداریم نیز علامت ? می‌آید.

نکته دیگری که برای دستورات وجود دارد این است که قبل از وارد شدن دستور، عبارت GET یا POST یا DELETE یا PUT وارد می‌شود که به این شکل دستورات به نحوی از هم جدا می‌شوند. دستوراتی که برای دریافت اطلاعات از سامانه استفاده می‌شوند در دسته GET قرار می‌گیرند؛ برای وارد کردن اطلاعات از دسته POST، برای تغییر اطلاعات موجود از دسته PUT و برای حذف اطلاعات از دسته DELETE استفاده می‌کنیم.

این نام‌گذاری دستورها در فازهای آتی پروژه که برنامه خود را روی وب عرضه خواهید کرد معنای خاص پیدا خواهند کرد. همچنین دقت کنید که ممکن است دو دستور با نام‌های مشابه وجود داشته باشند اما در دسته‌های متفاوتی قرار بگیرند، در این صورت ماهیت این دو دستور متفاوت بوده و در صورت فراخوانی آن‌ها، نتایج متفاوتی را مشاهده خواهیم کرد.

همینطور دقت داشته باشید که تمامی دستورها پس از اجرا شدن دارای خروجی مشخص هستند که منحصرا ذکر می‌شود. اگر در دستورات وارد شده کاربر، خطایی وجود داشته باشد، چه در دستورات چه در آرگومان‌ها، باید با توجه به توضیحاتی که همراه با هر دستور آمده است، خطای آن را خروجی دهید.

خروجی پروژه شما به صورت خودکار آزموده می‌شود؛ بنابراین خروجی شما باید دقیقاً همانند خروجی خواسته شده باشد. در غیر این صورت نمره‌ی بخش آزمون را از دست خواهید داد.

---

[4] command line

[5] ID

پاسخ دستورات

به ازای هر دستوری که اقدام به اجرای آن می‌کنیم، پاسخی از سمت سامانه دریافت می‌کنیم. این پاسخ می‌تواند اطلاعاتی که از سامانه خواسته شده است، باشد. اما گونه‌های دیگری از پاسخ نیز وجود دارد که در ادامه توضیح داده خواهد شد (رنگ‌های نمونه صرفا برای خوانایی می‌باشد و نباید از آن‌ها در خروجی‌تان استفاده کنید).

پاسخ درخواست موفقیت‌آمیز

اگر دستوری که کاربر وارد می‌کند به درستی انجام شود و به اتمام برسد، این پاسخ نمایش داده می‌شود (در برخی از حالات ممکن است خود دستور خروجی مفصل‌تری داشته باشد که در این صورت این پاسخ نمایش داده نمی‌شود، این حالات در ادامه و در توضیح هر بخش توضیح داده شده‌اند).

خروجی
OK

پاسخ خالی بودن

در صورتی که لیست درخواست‌شده از سامانه هیچ مورد قابل نمایشی نداشته باشد، این پاسخ به کاربر نمایش داده خواهد شد.

خروجی
Empty

پاسخ عدم وجود

در صورتی که دستور وارد شده در لیست دستورات GET, POST, DELETE, PUT وجود نداشت، این پیغام نمایش داده می‌شود. همچنین در صورتی که شناسه وجود نداشته باشد و به طور کلی در هر قسمت که جستجویی انجام می‌شود اما نتیجه‌ای در بر ندارد، این پاسخ داده می‌شود.

خروجی
Not Found

پاسخ درخواست اشتباه

اگر اولین قسمت ورودی کاربر، هیچ کدام از دستورهای PUT، POST، GET و DELETE نباشد، این پاسخ نمایش داده می‌شود. همچنین اگر دستور واردشده، اطلاعات کافی برای اجرا را در خود نداشته باشد و یا قالب دستور واردشده، با هیچ کدام از دستوراتی که در ادامه می‌آیند مطابقت نداشته باشد (آرگومان‌های دستور به درستی داده نشده باشند یا مقادیر آرگومان‌ها مطابق انتظار نباشند)، این پاسخ نمایش داده می‌شود. توجه داشته باشید که اگر مقادیر آرگومان‌های مورد انتظار درست بود اما تعدادی آرگومان بیشتر نیز داشتیم خطایی دریافت نمی‌شود.

خروجی
`Bad Request`

پاسخ عدم دسترسی (دسترسی غیرمجاز)

اگر یک کاربر دستوری وارد کرد که اجازه دسترسی به آن وجود نداشت این پیغام نمایش داده می‌شود.

خروجی
Permission Denied

سنجش خطاها

اولیت‌بندی سنجش خطاها در اجرای برنامه به صورت زیر می‌باشد:

1. ابتدا بررسی می‌شود که دستور با یکی از متدهای PUT،POST، GET یا DELETE شروع می‌شود. در صورتی که در ابتدای دستور وارد شده یکی از این چهار کلمه نباشد خطای Bad Request نمایش داده می‌شود.
2. پس از آن بررسی می‌شود که دستور وارد شده در لیست دستورات وجود دارد یا خیر؛ برای مثال دستور `GET something_non_existant` در دستورات برنامه نیست. در این حالت باید پاسخ Not Found نمایش داده شود.
3. سپس برای هر دستور اجازه دسترسی بررسی شود که در صورت عدم دسترسی با دستور Permission Denied مواجه شوند. در صورتی که کاربر هنوز لاگین نکرده باشد، یا در صورتی که یک کاربر دستورات یک نوع کاربر دیگر را وارد کند، از مصادیق این خطا هستند.
4. پس از آن حالات خاص هر دستور بررسی می‌شود، تضمین می‌شود در این حالت صرفا با یکی از حالات خاص مواجه هستیم (چند خطا در اینجا رخ نمی‌دهد).

دستورات

همه مقادیر داخل دستورات همگی داخل " "[6] قرار گرفته است. همچنین تضمین می‌شود مقدار هیچ آرگومانی شامل " نمی‌شود.

ثبت‌نام

این دستور به کاربران امکان می‌دهد تا در سامانه ثبت‌نام کنند. هر فرد با استفاده از یک نام کاربری یکتا در سامانه ثبت‌نام می‌کند. در صورتی که ثبت‌نام موفقیت‌آمیز باشد، کاربر مستقیما وارد سامانه می‌شود.

خطاها

- اگر نام کاربری تکراری باشد: Bad Request
- اگر کاربر از قبل در سامانه لاگین کرده باشد: Permission Denied

ورودی
`POST signup ? username “username” password “password”`
خروجی
`OK | Bad Request | Permission Denied`
نمونه ورودی
`POST signup ? username ”low_mist” password ”meow”`
نمونه خروجی
`OK`

[6] Double Quotation

ورود به سامانه

اگر کاربری قبلا در سامانه ثبت‌نام کرده باشد، پیش از استفاده از امکانات سامانه باید وارد سامانه شود. در صورتی که کاربر وارد سامانه نشده باشد و هر یک از دستورات بخش‌های بعد را وارد کند با پاسخ Permission Denied مواجه می‌شود.

خطاها

- اگر نام کاربری که کاربر وارد می‌کند در سامانه وجود نداشته باشد، پاسخ Not Found در جواب به کاربر داده می‌شود.
- اگر کاربر رمز خود را اشتباه وارد کند در حالی که نام کاربری موجود باشد، پاسخ Permission Denied در جواب به کاربر داده می‌شود.
- همچنین اگر کاربری از قبل وارد سامانه شده بود با وارد کردن این دستور پاسخ Permission Denied دریافت می‌کند (برای استفاده از این دستور حتما باید از قبل logout کرده باشیم).

ورودی
`POST login ? username “<username>” password “<password>”`
خروجی
`OK | Bad Request | Permission Denied | Not Found`
نمونه ورودی اول
`POST login ? username ”low_mist” password ”meoow”`
نمونه خروجی اول
`Permission Denied`
نمونه ورودی دوم
`POST login ? username ”low_mist” password ”meow”`
نمونه خروجی دوم
`OK`

خروج از سامانه

شخصی که قبلا در سامانه وارد شده بود با وارد کردن این دستور از سامانه خارج می‌شود. پس از آن می‌تواند دوباره با دستور login به همین حساب کاربری یا یک حساب کاربری دیگر وارد شود.

خطاها

- در صورتی که کاربر وارد سامانه نشده باشد پیام Permission Denied نمایش داده خواهد شد.

ورودی
`POST logout ?`
خروجی
`OK | Permission Denied | Bad Request`
نمونه ورودی
`POST logout ?`
نمونه خروجی
`OK`

نمایش محله‌ها

این دستور، محله‌های موجود در سامانه را به کاربر نمایش می‌دهد. خروجی دستور، لیست نام محله‌ها به ترتیب حروف الفبا می‌باشد (برای مقایسه نام محله‌ها از اپراتور مقایسه آماده زبان C++ استفاده کنید). جلوی نام هر یک از آن‌ها، نام محله‌های همسایه‌ی آن محله، به ترتیب نزدیکی به محله مذکور آمده است.[7] همچنین این دستور دارای یک آرگومان اختیاری به نام محله می باشد که در صورت وارد شدن آن توسط کاربر، فقط اطلاعات مربوط به محله مورد نظر نمایش داده خواهد شد.

خطاها

- در صورتی که هیچ محله‌ای موجود نبود، پیغام Empty نمایش داده خواهد شد.
- اگر آرگومان محله موجود بود اما در فایل محله‌ها موجود نبود، خطای Not Found نمایش داده می‌شود.

ورودی
`GET districts ? district “<district>”`
خروجی

`<district1>: <neighbor_district11>, …, <neighbor_district{n-1}>`
`<district2>: <neighbor_district21>, …, <neighbor_district{n-1}>`
`… | Empty | Not Found | Bad Request`

نمونه ورودی اول
`GET districts ?`
نمونه خروجی اول

`Azadi: Behboodi, SatarKhan, Shadman`
`Behboodi: Azadi, SatarKhan, Shadman`
`SatarKhan: Shadman, Behboodi, Azadi`
`Shadman: SatarKhan, Behboodi, Azadi`

نمونه ورودی دوم
`GET districts ? district “SatarKhan”`
نمونه خروجی دوم
`SatarKhan: Shadman, Behboodi, Azadi`

[7] در صورت ابهام در مفهوم نزدیکی، به بخش فایل محله ها مراجعه نمایید.

تنظیم کردن مکان کنونی کاربر

با این دستور، مکان کنونی کاربر، برای نمایش رستوران‌های نزدیک به او در هنگام جستجوی غذا، تنظیم می‌شود.

خطاها

- این مکان باید یکی از محله‌هایی باشد که در داده‌های محله‌ها وجود دارد در غیر این صورت با خطای Not Found مواجه می شویم.

ورودی
`PUT my_district ? district “<district>”`
خروجی
`OK | Not Found | Bad Request`
نمونه ورودی اول
`PUT my_district ? district “Saadat Abad”`
نمونه خروجی اول
`OK`
نمونه ورودی دوم
`PUT my_district ? district “Merrikh”`
نمونه خروجی دوم
`Not Found`

مشاهده لیست رستوران‌ها

با وارد کردن این دستور، لیست تمام رستوران‌های موجود در سامانه، به ترتیب نزدیکی به کاربر نمایش داده می‌شوند. به این صورت که ابتدا رستوران‌های درون محله فعلی کاربر، سپس رستوران‌های محله‌های اطراف محله کاربر، سپس رستوران‌های محله‌های اطراف محله‌های اطراف محله کاربر و الی آخر نمایش داده می‌شوند، به طوری که تمام رستوران‌ها در خروجی می‌آیند. ترتیب بین محله‌ها بر اساس همان پارامتر نزدیکی تعریف شده در بخش‌های قبل بوده و رستوران‌های درون هر محله، بر حسب حروف الفبا مرتب می‌شوند[8].
این دستور یک آرگومان اختیاری به نام غذا هم دارد که در صورت وارد کردن آن توسط کاربر، باید فقط لیست رستوران‌هایی که آن غذا را ارائه می‌دهند، با همان ترتیب قبلی نمایش داده شود.

خطاها

- درصورتی که کاربر موقعیت مکانی خود را قبلا تنظیم نکرده باشد با خطای Not Found مواجه می‌شویم.
- اگر رستورانی وجود نداشت، باید پیغام Empty نمایش داده شود.

ورودی
`GET restaurants ? food_name "<food_name>"`
خروجی
`<restaurant_name1(district)>` `… | Empty | Not Found | Bad Request`
نمونه ورودی اول
`GET restaurants ?`
نمونه خروجی اول

`Apadana (Omid Town)`
`Bon Bon (Enghelab)`
`Nofel Loshato (Enghelab)`

نمونه ورودی دوم
`GET restaurants ? food_name "Kabab Barg"`
نمونه خروجی دوم
`Apadana (Omid Town)` `Nofel Loshato (Enghelab)`

[8] مشابه الگوریتم bfs

نمونه ورودی سوم

```
GET restaurants ? food_name “Sushi”
```

نمونه خروجی سوم

```
Empty
```

مشاهده اطلاعات یک رستوران

با وارد کردن این دستور، امکان مشاهده اطلاعات یک رستوران با استفاده از نام آن امکان‌پذیر است. اطلاعاتی که از رستوران نمایش داده می‌شود بدین ترتیب است:

- خط اول: نام رستوران
- خط دوم: نام محله‌ای که رستوران در آن قرار دارد
- خط سوم: ساعت کاری رستوران
- خط چهارم: لیستی شامل غذاهای موجود در منوی رستوران. غذاهای موجود در لیست با علامت کاما از هم جدا شده‌اند و ترتیب غذاها به ترتیب حروف الفباست. همچنین جلوی نام هر غذا داخل پرانتز قیمت آن غذا نوشته شده است
- در n خط بعدی شماره میزهای رستوران به ترتیب خواهد آمد که جلوی شماره هر میز، به تعداد رزروهای آن میز پرانتز وجود دارد که با علامت کاما از هم جدا شده‌اند ساختار هر پرانتز به شکل زیر است:

(ساعت پایان رزرو-ساعت آغاز رزرو)

توجه داشته باشید که در کل سامانه، ساعت ها فقط با فرمت یک عدد صحیح مثبت که نشان دهنده ساعت (از 1 تا 24) هستند نمایش داده می شوند و احتیاجی به وارد کردن، پردازش، و یا نمایش دقیقه نیست. همچنین هیچ یک از بازه‌های رزرو از ساعت 24 عبور نخواهند کرد. همچنین شماره هر میز در هر صورت نشان داده خواهد شد حتی اگر برای آن میز، هیچ رزروی صورت نگرفته باشد.

خطاها

- در صورتی که نام رستوران وجود نداشته باشد با خطای Not Found مواجه می شویم.

ورودی

```
GET restaurant_detail ? restaurant_name “<restaurant_name>”
```

خروجی

```
Name: <restaurant_name>
District: <restaurant_district_name>
Time: <start_time>-<end_time>
```

```
Menu: <food_name1>(<food_price1>), <food_name2>(<food_price2>)
<table_id1>: (<reserve_start11>,
<reserve_end11>),(<reserve_start12>, <reserve_end12>)
<table_id2>: (<reserve_start21>,
<reserve_end21>),(<reserve_start22>, <reserve_end22>)
… | Not Found | Bad Request
```

نمونه ورودی اول

```
GET restaurant_detail ? restaurant_name “Chicken Family”
```

نمونه خروجی اول

```
Name: Chicken Family
District: Yousef Abad
Time: 12-22
Menu: Sib Zamini(70000), Sokhari(200000)
1: (14-16), (17-19)
2:
3:
4: (21-22)
```

نمونه ورودی دوم

```
GET restaurant_detail ? restaurant_name “Na Koja Abad”
```

نمونه خروجی دوم

```
Not Found
```

رزرو رستوران

با وارد کردن این دستور کاربر یکی از میزهای یک رستوران را برای یک بازه زمانی رزرو می کند. این دستور همچنین یک آرگومان اختیاری برای سفارش غذا دارد. در صورتی که کاربر تمایل به سفارش غذا نداشته و فقط قصد رزرو میز را داشته باشد، آرگومان اختیاری را وارد نمی‌کند. به هر رزرو کاربر یک شناسه اختصاص می‌یابد که در هر رستوران از 1 شروع شده و به ازای هر رزرو یکی افزایش پیدا می‌کند (شناسه‌های رزروها در سطح رستوران یکتا خواهند بود، اما یک کاربر می‌تواند دو رزرو با شناسه‌های یکسان اما در رستوران‌های متفاوت داشته باشد). در خروجی این دستور علاوه بر خلاصه‌ای از اطلاعات رزرو، مجموع قیمت غذاهای انتخابی توسط کاربر نیز نمایش داده می‌شود. در صورتی که کاربر غذایی سفارش نداده باشد قیمت نمایشی برابر صفر است. توجه داشته باشید که یک غذا ممکن است چند بار در لیست سفارش باشد و غذاها با کاما از هم جدا شده‌اند (غذاها می‌توانند شامل کاراکتر فاصله باشند اما در میان آن‌ها نباید کاراکتر فاصله داشته باشیم و تنها کاراکتر کاما آن‌ها را از هم جدا می‌کند).

فرض کنید در یک رستوران هیچ رزروی ثبت نشده است؛ اگر فردی رزروی در آن رستوران ثبت کند، شناسه آن رزرو برابر یک می‌شود، حال اگر فردی دیگر یا همین فرد در همان رستوران رزروی را ثبت کند، شناسه آن رزرو برابر دو می‌شود.

خطاها

- اگر بازه زمانی انتخاب‌شده توسط کاربر برای یک میز در رستوران با بازه‌های رزرو شده آن میز تداخل داشت، با خطای Permission Denied مواجه می‌شویم.
- اگر کاربر در آن زمان در همان رستوران یا یک رستوران دیگر رزروی داشت که با این رزرو تداخل داشت (تداخل به معنی این است که ساعت پایان یک رزرو بین ساعت آغاز و پایان یک رزرو دیگر باشد، به عبارت دیگر بازه 21-19 با بازه 22-21 تداخل ندارد اما با 23-20 تداخل دارد)، با خطای Permission Denied مواجه می‌شویم.
- اگر ساعت رزرو خارج از محدوده ساعت کاری رستوران یا از 1 تا 24 بود، با خطای Permission Denied مواجه می‌شویم.
- اگر غذای خواسته شده در رستوران موجود نبود و یا میز خواسته شده در رستوران وجود نداشت یا اگر رستوران در سامانه موجود نبود، با خطای Not Found مواجه می شویم.

ورودی

```
POST reserve ? restaurant_name "<restaurant_name>" table_id "<table_id>" start_time "<start_time>" end_time "<end_time>" foods "<food_names> "
```

خروجی

```
Reserve ID: <reserve_id>
```

```
Table <table_id> for <start_time> to <end_time> in
<restaurant_name>
Price: <price_of_foods> |
Bad Request | Permission Denied | Not Found
```

نمونه ورودی اول

```
POST reserve ? restaurant_name “Chicken Family” table_id ”3”
start_time ”16” end_time ”17” foods “Sib Zamini,Sokhari”
```

نمونه خروجی اول

```
Reserve ID: 3
Table 3 for 16 to 17 in Chicken Family
Price: 170000
```

نمونه ورودی دوم

```
POST reserve ? restaurant_name “Chicken Family” table_id ”1”
start_time ”15” end_time ”17” foods “Sib Zamini,Sokhari”
```

نمونه خروجی دوم

```
Permission Denied
```

نمایش رزروهای انجام شده توسط کاربر

با استفاده از این دستور، لیستی از تمام رزروهایی که کاربر در سامانه دارد نمایش داده می‌شود. برای هر رزرو، باید نام رستوران، شماره میز، زمان شروع و زمان پایان رزرو و غذاهایی که رزرو شده (در صورت وجود) به همراه تعداد آنها را نمایش داد. ترتیب نمایش خروجی، بر حسب زمان رزروها می‌باشد. این دستور همچنین می‌تواند شامل دو آرگومان اختیاری می‌باشد. آرگومان اول، "نام رستوران" است. اگر نام رستوران وارد شود، رزروهای کاربر در رستوران مشخص‌شده چاپ می‌شود. آرگومان دوم، "شناسه رزرو" است. این آرگومان حتما باید به همراه آرگومان "نام رستوران" همراه باشد و زمانی که وارد شود، اطلاعات رزرو با شناسه داده شده در رستوران مد نظر نمایش داده خواهد شد.

خطاها

- اگر رزروی وجود نداشت باید پیغام Empty چاپ شود.
- اگر شناسه رزرو داده وارد شود اما رستوران مشخص نشود،خطای Bad Request رخ می‌دهد.
- اگر نام رستوران و شناسه رزرو وارد شود اما رزرو مورد نظر متعلق به کاربر فعلی نباشد، خطای Permission Denied رخ می‌دهد.
- اگر نام رستوران و شناسه رزرو وارد شود اما رزرو مورد نظر در لیست رزروهای رستوران موجود نباشد، خطای Not Found رخ می‌دهد.
-

ورودی

```
GET reserves ? restaurant_name ”<restaurant_name>” reserve_id
”<reserve_id>”
```

خروجی

```
<reserve_id1>: <restaurant_name1> <table_id1>
<start_time1>-<end_time1> <food11(number11)> <food12(number12)>
<reserve_id1>: <restaurant_name2> <table_id2>
<start_time2>-<end_time2> <food21(number21)> <food22(number(22)>
… | Permission Denied | Bad Request | Empty
```

نمونه ورودی اول

```
GET reserves ?
```

نمونه خروجی اول

```
1: Gotham 12 18-19
```

```
1: Espinas Pallas 7 20-21 Steak(1) French Fries(1)
3: Espinas Pallas 9 21-23 Pizza(3)
```

نمونه ورودی دوم

```
GET reserves ? restaurant_name "Espinas Pallas"
```

نمونه خروجی دوم

```
1: Espinas Pallas 7 20-21 Steak(1) French Fries(1)
3: Espinas Pallas 9 21-23 Pizza(3)
```

نمونه ورودی سوم

```
GET reserves ? restaurant_name "Espinas Pallas" reserve_id "3"
```

نمونه خروجی سوم

```
3: Espinas Pallas 9 21-23 Pizza(3)
```

نمونه ورودی چهارم

```
GET reserves ? restaurant_name "Espinas Pallas" reserve_id "5"
```

نمونه خروجی چهارم

```
Permission Denied | Not Found[9]
```

[9] با توجه به شرایط، یکی از این خروجی‌ها می‌تواند معتبر باشد

تغییر اطلاعات رزرو

با وارد کردن این دستور کاربر می‌تواند اطلاعات رزرو ثبت شده توسط خود را تغییر دهد. کاربر باید در ابتدا شناسه رزرو و نام رستورانی که می‌خواهد آن را تغییر دهد وارد کند. سپس باید آرگومان‌هایی که قرار است تغییر کنند را وارد کند. دقت داشته باشید که اگر آرگومانی در این دستور مقداردهی نشده بود آن آرگومان تغییری نمی‌کند و صرفا آرگومان‌هایی که مشخص شده‌اند تغییر می‌کنند.

خطاها

- اگر نام رستوران یا شناسه رزرو در آن رستوران موجود نبود با خطای Not Found مواجه می شویم.
- اگر شناسه رزرو وارد شده در رستوران مورد نظر مربوط با کاربر فعلی نباشد، با خطای Permission Denied مواجه می‌شویم.
- اگر شناسه رزرو در رستوران مورد نظر موجود نباشد، با خطای Not Found مواجه می شویم.
- اگر بازه زمانی انتخاب‌شده توسط کاربر برای یک میز در رستوران با بازه‌های رزرو شده آن میز تداخل داشت، با خطای Permission Denied مواجه می‌شویم (دقت داشته باشید که اگر رزرو ما مثلا برای ساعت 21-19 بود و می‌خواستیم آن را به 22-20 تغییر دهیم نباید با خطایی مواجه شویم).
- اگر ساعت رزرو خارج از محدوده ساعت کاری رستوران یا از 1 تا 24 بود، با خطای Permission Denied مواجه می شویم.
- اگر غذای خواسته شده در رستوران موجود نبود و یا میز خواسته شده در رستوران وجود نداشت یا اگر رستوران در سامانه موجود نبود، با خطای Not Found مواجه می شویم.

ورودی

```
PUT edit_reserve ? reserve_id "<reserve_id>" restaurant_name "<restaurant_name>" start_time "<start_time>" end_time "<end_time>" foods "<food_names>"
```

خروجی

```
OK | Bad Request | Permission Denied | Not Found
```

نمونه ورودی اول

```
POST edit_reserve ? reserve_id "3" restaurant_name "maman joon" table_id "3" start_time "16" end_time "17"
```

نمونه خروجی اول

```
OK
```

حذف رزرو

کاربر با این دستور می‌تواند رزروی که قبلا انجام داده بود را حذف کند. برای این کار، کاربر باید شناسه رزرو و نام رستوران مورد نظر را وارد کند. توجه داشته باشید پس از حذف یک رزرو اگر کاربر یک رزرو جدید بخواهد انجام دهد باید به همان روال قبل شناسه جدید به رزرو تعلق بگیرد، یعنی به طور مثال اگر کاربر رزرو سوم مربوط به یک رستوران که تا به الان پنج رزرو داشته را حذف کرد، آنگاه رزرو بعدی در آن رستوران شناسه شش خواهد گرفت (همچنین اگر رستورانی تا به الان 2 رزرو داشته و رزرو دوم حذف شود، رزرو بعدی بدون توجه به این حذف شناسه 3 خواهد گرفت).

خطاها

- اگر نام رستوران یا شناسه رزرو در آن رستوران موجود نبود با خطای Not Found مواجه می شویم.
- اگر شناسه رزرو وارد شده در رستوران مورد نظر مربوط با کاربر فعلی نباشد، با خطای Permission Denied مواجه می‌شویم.

ورودی
`DELETE reserve ? restaurant_name "<restaurant_name>" reserve_id "<reserve_id>"`
خروجی
`OK | Not Found | Bad Request`
نمونه ورودی اول
`DELETE reserve ? restaurant_name "Haida" reserve_id "3"`
نمونه خروجی اول
`OK`
نمونه ورودی دوم
`DELETE reserve ? restaurant_name "Haida" reserve_id "6785"`
نمونه خروجی دوم
`Not Found`

نکات و نحوه تحویل

- برای تحویل این پروژه، لازم است کد خود را در یک مخزن[10] در GitHub بارگذاری کنید و سپس لینک مخزن به همراه Hash آخرین کامیت[11] را در صفحه eLearn درس بارگذاری نمایید.
نمونه متن خواسته شده در سامانه ای‌لرن (سه بخش <username> و <repository_name> و <last_commit_hash> را جایگزین کنید):

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

- دقت کنید که پروژه شما باید Multi-file باشد و Makefile داشته باشد. همین‌طور در Makefile خود مشخص کنید که از استاندارد c++20 استفاده می‌کنید.
- فایل‌های خود را در مخزنی به نام AP-F03-A7-<SID> قرار 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. دقت داشته باشید که برای هر سه فاز از این مخزن استفاده خواهید کرد و همه تغییرات خود را در شاخه[12] main انجام دهید و نیازی به پیاده‌سازی شاخه‌ها نیست (هر چند برای کارایی بهتر توصیه می‌شود انجام دهید ولی در پایان تغییرات خود را به شاخه main نیز منتقل کنید). دقت داشته باشید که چون تحویل هر سه فاز با یکدیگر خواهد بود مشخصات کامیت خود را در هر فاز به درستی وارد کنید.
- با توجه به حجم نسبتا زیاد این فاز از تمرین توصیه می‌شود قبل از پیاده‌سازی کد طراحی اولیه‌ای برای منطق برنامه و روندهای آن مثل ثبت‌نام و ... انجام دهید و پس از این طراحی شروع به پیاده‌سازی آن کنید. از آن جایی که در فازهای بعدی شما باید رابط کاربری برنامه‌ی خود را از command-line به روش‌هایی دیگر تغییر دهید، بهتر است تا طراحی برنامه‌ی شما طوری باشد که کمترین وابستگی میان منطق برنامه و رابط کاربری آن وجود داشته باشد.
- 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.
- درستی برنامه شما از طریق آزمون‌های خودکار سنجیده می‌شود؛ بنابراین از درستی کامل قالب خروجی برنامه خود اطمینان حاصل کنید و از دادن خروجی‌هایی که در صورت پروژه ذکر نشده است اجتناب کنید.
- دقت کنید که نام فایل اجرایی شما باید UTaste (بدون هیچ پسوندی مانند exe یا out) باشد.
- سوالات خود را تا حد ممکن در فروم درس مطرح کنید تا سایر دانشجویان نیز از پاسخ آن‌ها بهره‌مند شوند. در صورتی که قصد مطرح کردن سوال خاص‌تری داشتید، از طریق ایمیل با طراحان این فاز پروژه ارتباط برقرار کنید.

---

[10] Repository
[11] Commit
[12] Branch

- توجه داشته باشید که حالت‌های خاصی که در صورت پروژه ذکر نشده است در تست‌های خودکار نخواهد بود و می‌توانید به هر شکلی که مد نظر دارید آن‌ها را مدیریت کنید.
- 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، فایل اجرایی و آپلود موارد خواسته شده** باعث کسر 5 درصد از نمره شما خواهد شد.

نمرات

- تمیزی کد
  - رعایت کردن نام‌گذاری صحیح و انسجام[13]
  - عدم وجود کد تکراری
  - رعایت دندانه‌گذاری[14]
  - عدم استفاده از متغیرهای گلوبال
  - استفاده **صحیح** از متغیرهای ثابت[15] به جای Magic Value-ها
  - ساختاردهی کد در قالب توابع کوتاه که فقط یک کار را انجام می‌دهند
- درستی کد
  - آزمون‌های خودکار
  - پیاده‌سازی صحیح کارکردهای خواسته شده
- طراحی
  - طراحی صحیح و منطقی در شی‌گرایی و ارث‌بری
  - رعایت Encapsulation
  - جداسازی منطق کد از ورودی/خروجی و استفاده از کلاس جداگانه برای مدیریت دستورات
  - استفاده مناسب از استثناها برای مدیریت خطا
  - میک‌فایل و چندفایلی

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

[13] Consistency

[14] Indentation

[15] Constant