برنامه‌سازی پیشرفته

تمرین کامپیوتری شماره ۷

مدرس: رامتین خسروی

طراحان: هادی بابالو، علی عطاءاللهی، علی حمزه پور، محمد صادقی، عرفان عسگری، شهریار عطار

مهلت تحویل: جمعه ۸ دی ۱۴۰۲، ساعت ۲۳:۵۵

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 انتهای این پروژه شما باید یک سایت اشتراک‌گذاری آهنگ را پیاده‌سازی کنید؛ هدف اصلی این سایت ایجاد امکان به اشتراک‌گذاری آهنگ توسط هنرمندان مختلف است. پیشنهاد آهنگ‌های مرتبط به کاربران، امکان ساخت پلی‌لیست، امکان جستجوی آهنگ‌ها، امکان اضافه کردن آهنگ‌ها و ... از معدود امکاناتی است که در طول فازهای مختلف آن‌ها را پیاده‌سازی خواهید کرد.

نکته قابل توجه در این پروژه این است که بهتر است پروژه به صورت قسمت به قسمت پیاده‌سازی و تست شود. به طوری که ابتدا یک ساختار کلی از پروژه پیاده‌سازی شود و سپس دستورات مختلف به آن اضافه گردد.

شرح تمرین

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[1]، 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[2] 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 سیستم 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.

---

[1] command line

[2] ID

پاسخ دستورات

به ازای هر دستوری که اقدام به اجرای آن می‌کنیم، پاسخی از سمت سیستم دریافت می‌کنیم. این پاسخ می‌تواند اطلاعاتی که از سیستم خواسته شده است، مانند اطلاعات یک آهنگ یا لیست کاربران 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 شده وارد در لیست دستورات GET, POST, DELETE, PUT وجود نداشت، این پیغام نمایش داده می‌شود. همچنین در صورتی که شناسه آهنگ یا کاربر مربوطه 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 لیست 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؛ مثلا برای سال ساخت آهنگ یک مقدار غیر عددی دهیم)، این پاسخ نمایش داده می‌شود.

خروجی
Bad Request

پاسخ عدم دسترسی (دسترسی غیرمجاز)

اگر یک کاربر دستوری از لیست دستورات دیگر کاربران را وارد کرد (مثلا کاربر عادی دستورات هنرمند را وارد کند یا برعکس) 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 کاربر عادی دستورات کاربر هنرمند را وارد کند یا برعکس از مصادیق این خطا می‌باشند.
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

دستورات مشترک کاربر عادی و هنرمند

توجه کنید که در همه دستورات مقدار همه آرگومان‌ها داخل <> نوشته می‌شود؛ این موضوع برای اینکه آرگومان‌ها بتوانند کاراکتر اسپیس را هم شامل شوند نیاز است. با این حال دقت کنید که در نمونه‌های داده شده، مقدار آرگومان‌ها به صورت {arg_value} نشان داده می‌شوند؛ این یعنی مقدار اصلی arg_value در ورودی داده می‌شود و کاراکترهای { و } در ورودی اصلی ظاهر نخواهند شد. برای درک بهتر این موضوع به نمونه‌های ورودی داده‌شده در هر بخش توجه کنید.

ثبت‌نام

با استفاده از این دستور، افراد می‌توانند در سامانه به عنوان کاربر عادی (user) و یا هنرمند (artist) ثبت‌نام کنند.

- هر فرد باید با استفاده از یک نام کاربری یکتا در سامانه ثبت‌نام را انجام دهد.
- در هنگام ثبت‌نامِ هر کاربرِ جدید یک شناسه یکتا به این کاربر اختصاص داده می‌شود. این شناسه‌ها از یک شروع شده و به ازای هر کاربر جدیدی که در سامانه ثبت‌نام می‌کند، یک واحد افزایش می‌یابد. یعنی اولین کاربر ثبت‌نام‌شده دارای شناسه یک، دومین کاربر ثبت‌نام شده دارای شناسه دو و ... است.
- در صورتی که ثبت‌نام موفق‌آمیز باشد، کاربر مستقیما وارد سامانه می‌شود.
- مقدار پارامتر mode نشان‌دهنده نوع کاربر می‌باشد که یا artist می‌باشد یا user.
- اگر نام کاربری در سیستم موجود باشد، درخواست معتبر نبوده و در جواب آن پاسخ Bad Request داده می‌شود. همچنین اگر مقدار آرگومان mode چیزی غیر از artist یا user بود نیز این پاسخ داده می‌شود.
- همچنین اگر کاربری از قبل وارد سیستم 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 signup ? username <{username}> password <{password}> mode
<mode>
```

خروجی

```
OK | Bad Request
```

نمونه ورودی

```
POST signup ? username <Misagh> password <meow> mode <artist>
```

نمونه خروجی

```
OK
```

ورود به سیستم

اگر کاربری قبلا در سامانه ثبت‌نام کرده باشد، پیش از استفاده از امکانات سامانه باید وارد سیستم شود.

- 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 وجود داشت، پاسخ Permission Denied در جواب به کاربر داده می‌شود.
- در صورتی که کاربر وارد سامانه نشده باشد و هر یک از دستورهای بخش‌های بعد را وارد کند با پاسخ Permission Denied مواجه می‌شود.
- همچنین اگر کاربری از قبل وارد سیستم 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 <Misagh> password <meoow>
```

نمونه خروجی اول

```
Permission Denied
```

نمونه ورودی دوم

```
POST login ? username <Misagh> password <meow>
```

نمونه خروجی دوم

```
OK
```

خروج از سیستم

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

ورودی

```
POST logout ?
```

خروجی

```
OK | Permission Denied | Bad Request
```

نمونه ورودی

```
POST logout ?
```

نمونه خروجی

```
OK
```

نمایش آهنگ‌های موجود

کاربر می‌تواند با دستور زیر فهرست آهنگ‌های موجود به همراه اطلاعات خلاصه‌شده آن‌ها که شامل شناسه آهنگ، نام آهنگ و نام هنرمند آن است را مشاهده کند.

- در خروجی این دستور، خط اول ترتیب ستون‌ها بوده و بعد از آن در هر خط اطلاعات مربوط به یک آهنگ نمایش داده می‌شود. هر کدام از این مقادیر در خروجی با "," از هم جدا شده‌اند. ترتیب ستون‌ها مانند نمونه خروجی می‌باشد.
- خروجی بر اساس شناسه آهنگ‌ها، به صورت صعودی مرتب شده است.
- در صورتی که آهنگی وجود نداشت پاسخ Empty نمایش داده می‌شود (در این حالت خط اول که ترتیب ستون‌ها را مشخص می‌کند نیز نمایش داده نمی‌شود).

ورودی

```
GET musics ?
```

خروجی

```
ID, Name, Artist
{id_1}, {name_1}, {artist_1}
```

```
{id_2}, {name_2}, {artist_2}
… | Permission Denied | Bad Request | Empty
```

نمونه ورودی

```
GET musics ?
```

نمونه خروجی

```
ID, Name, Artist
1, Flying, Anathema
2, Nothing Else Matters, Metallica
5, Said & Done, Bad Omens
```

نمایش مشخصات یک آهنگ

کاربر می‌تواند با وارد کردن شناسه یک آهنگ به عنوان یک آرگومان، اطلاعات کامل یک آهنگ را دریافت کند.

- در خط اول خروجی ترتیب ستون‌ها می‌آید. ترتیب این ستون‌ها مانند نمونه خروجی می‌باشد.
- اطلاعات کامل آهنگ شامل شناسه آهنگ، نام آهنگ، نام هنرمند، سال ساخت آهنگ، نام آلبوم، تگ‌های آهنگ و مدت زمان آهنگ می‌شود. تگ‌های آهنگ در صورت وجود با ";" از هم جدا شده‌اند.
- در صورتی که شناسه مورد نظر در آهنگ ها نبود پاسخ Not Found نمایش داده می‌شود.

ورودی

```
GET musics ? id <{id}>
```

خروجی

```
ID, Name, Artist, Year, Album, Tags, Duration
{id}, {name}, {artist}, {year}, {album}, {tags}, {duration}
… | Not Found | Bad Request
```

نمونه ورودی

```
GET musics ? id <4>
```

نمونه خروجی

```
ID, Name, Artist, Year, Album, Tags, Duration
4, Zendegi, Hayedeh, 1972, Zendegi, Persian;Nostalgic, 5:15
```

نمایش کاربران موجود در سامانه

کاربر با استفاده از این دستور می‌تواند فهرست کاربران موجود در سامانه به همراه اطلاعات خلاصه‌شده آن‌ها را مشاهده کند.

- در خط اول خروجی ترتیب ستون‌ها می‌آد. ترتیب این ستون‌ها مانند نمونه خروجی می‌باشد.
- اطلاعات نمایش‌داده‌شده در این بخش در صورتی که کاربر مورد نمایش عادی باشد، شامل شناسه کاربر، نوع کاربر، نام کاربر و تعداد پلی‌لیست‌های ساخته‌شده توسط کاربر است.
- اطلاعات نمایش‌داده‌شده در صورتی که کاربرِ مورد نمایش هنرمند باشد، شامل شناسه کاربر، نوع کاربر، نام کاربر و تعداد آهنگ‌های به‌اشتراک‌گذاشته شده توسط هنرمند است.
- توجه شود که در هر دو بخش اسامی به ترتیب شناسه کاربری (به صورت صعودی) نمایش داده می‌شوند.

ورودی

```
GET users ?
```

خروجی

```
ID, Mode, Username, Playlists_number/Songs_number
{id_1}, {mode_1}, {username_1}, {number_1}
{id_2}, {mode_2}, {username_2}, {number_2}
… | Permission Denied | Bad Request | Empty
```

نمونه ورودی

```
GET users ?
```

نمونه خروجی

```
ID, Mode, Username, Playlists_number/Songs_number
1, user, Taraneh, 5
2, artist, Ebi, 2
3, user, Patrick, 4
```

نمایش یک کاربر در سامانه

کاربر با استفاده از این دستور می‌تواند اطلاعات کامل یک کاربر موجود در سامانه را مشاهده کند.

- اطلاعات نمایش داده شده شامل شناسه، نوع کاربر، اسم و همچنین برای کاربران عادی پلی‌لیست‌های آن‌ها و برای هنرمندان، آهنگ‌های آن‌ها می‌باشد.

- دقت کنید که در بخش Playlists/Songs در صورتی که کاربر مورد نمایش عادی بود باید Playlists و در صورتی که کاربر مورد نمایش هنرمند بود باید Songs چاپ شود. برای درک بهتر به تفاوت دو نمونه داده‌شده توجه کنید.

ورودی

```
GET users ? id <{id}>
```

خروجی

```
ID: {id}
Mode: {mode}
Username: {username}
Playlists/Songs: {playlist_song_name1}, {playlist_song_name2}, …
| Not Found
```

نمونه ورودی اول

```
GET users ? id <2>
```

نمونه خروجی اول

```
ID: 2
Mode: user
Username: FabulousMatin
Playlists: Jazz playlist, Road playlist
```

نمونه ورودی دوم

```
GET users ? id <1>
```

نمونه خروجی دوم

```
ID: 1
Mode: artist
Username: The Cranberries
Songs: Zombie, Linger, Ode to My Family
```

دستورات کاربر عادی

ایجاد پلی‌لیست

کاربر می‌تواند با وارد کردن این دستور یک پلی‌لیست جدید با نام دلخواه خود بسازد. نام پلی‌لیست نیز به عنوان آرگومان داده می‌شود. دقت کنید که نام پلی‌لیست در سطح کاربران یکتا می‌باشد (دو کاربر مختلف ممکن است دو پلی‌لیست با نام‌های مشابه داشته باشند ولی یک کاربر دو پلی‌لیست با نام‌های مشابه نخواهد داشت).

- در صورتی که این کاربر پلی‌لیستی با همین نام قبلا ساخته بود با پاسخ Bad Request مواجه می‌شویم.

ورودی

```
POST playlist ? name <{name}>
```

خروجی

```
OK | Permission Denied | Bad Request
```

نمونه ورودی

```
POST playlist ? name <Rock n Roll>
```

نمونه خروجی

```
OK
```

اضافه کردن آهنگ به پلی‌لیست

کاربر با استفاده از این دستور می‌تواند با وارد کردن نام یکی از پلی‌لیست‌های متعلق به خود و همچنین شناسه یک آهنگ، آن آهنگ را به پلی‌لیست مربوطه اضافه کند.

- در صورتی که اسم پلی‌لیست در بین پلی‌لیست‌های کاربر نبود یا شناسه آهنگ در بین آهنگ‌ها موجود نبود پاسخ Not Found نشان داده شود.

ورودی

```
PUT add_playlist ? name <{name}> id <{id}>
```

خروجی

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

نمونه ورودی

```
PUT add_playlist ? name <Rock n Roll> id <11>
```

نمونه خروجی

```
OK
```

نمایش پلی‌لیست‌های متعلق به یک کاربر

کاربر با استفاده از این دستور و با وارد کردن شناسه یک کاربر می‌تواند اطلاعات پلی‌لیست‌های او را مشاهده کند.

- ترتیب نمایش پلی‌لیست‌ها در خروجی این دستور به ترتیب حروف الفبا (مرتب‌سازی بر اساس ASCII انجام دهید) با توجه به اسم پلی‌لیست‌ها است.
- در خط اول خروجی نام ستون‌ها نمایش داده می‌شوند. در ادامه خروجی باید شناسه، نام، تعداد آهنگ‌ها و مدت زمان کل پلی‌لیست نمایش داده شود؛ مدت زمان کل آن برابر مجموع مدت زمان آهنگ‌های داخل آن است.
- فرمت زمان کل پلی‌لیست به صورت HH:MM:SS است، اگر پلی‌لیستی نیز به طور مثال زمانش 45:54 بود به این صورت نمایش داده می‌شود 00:45:54، یعنی همواره همه این ارقام باید چاپ شوند، حتی اگر صفر باشند.
- یک کاربر می‌تواند شناسه خودش را وارد کند تا پلی‌لیست‌های خودش را دریافت کند.
- در صورتی که شناسه کاربر در سامانه موجود نباشد، پاسخ Not Found به کاربر نمایش داده می‌شود.
- در صورتی که شناسه کاربر متعلق به یک هنرمند بود با پاسخ Bad Request مواجه می‌شویم.

ورودی

```
GET playlist ? id <{id}>
```

خروجی

```
Playlist_name, Songs_number, Duration
{playlist_name_1}, {songs_number_1}, {duration}
{playlist_name_2}, {songs_number_2}, {duration}
… | Not Found | Bad Request | Permission Denied
```

نمونه ورودی

```
GET playlist ? id <8>
```

نمونه خروجی

```
Playlist_name, Songs_number, Duration
Rock n Roll, 66, 01:44:33
Wrapped 2023, 50, 00:55:22
```

جستجوی آهنگ

با استفاده از این دستور کاربر می‌تواند شناسه آهنگ مورد نظر خود را پیدا کند.

- برای جستجو سه پارامتر اختیاری داریم، در صورت وجود هر پارامتر در صورتی آهنگ در خروجی نمایش داده می‌شود که با همه پارامترها تطابق داشته باشد؛ توجه کنید که تطابق باید نسبت به حروف بزرگ و کوچک حساس باشد.
- مقادیر هر یک از سه آرگومان ورودی به صورت رشته داده می‌شوند. برای آرگومان‌های name و artist، تمام آهنگ‌هایی که اطلاعات آن‌ها شامل زیررشته آرگومان ورودی هستند، منطبق حساب می‌شوند. به طور مثال اگر برای آرگومان artist، رشته park ورودی داده شود، تمام آهنگ‌های به اشتراک گذاشته شده توسط linkin park باید در خروجی نمایش داده شوند.
- برای آرگومان tag در صورتی که رشته ورودی جزو tag-های یک آهنگ باشد، آن آهنگ منطبق بوده و در خروجی نمایش داده می‌شود.
- در خروجی نیز صرفا شناسه و نام همه آهنگ‌هایی که قابل قبول هستند نمایش داده می‌شود. ترتیب نمایش آهنگ‌ها نیز بر اساس شناسه است.
- در صورتی که لیست آهنگ‌های منطبق خالی بود پیام Empty چاپ شود.

ورودی

```
GET search_music ? name <{name}> artist <{name}> tag <{tag}>
```

خروجی

```
ID, Name, Artist
{id}, {name}, {artist}
… | Empty | Bad Request | Permission Denied
```

نمونه ورودی

```
GET search_music ? name <ing> artist <Park>
```

نمونه خروجی

```
ID, Name, Artist
2, Crawling, Linkin Park
5, Talking To Myself, Linkin Park
9, Burning In The Skies, Linkin Park
```

دستورات کاربر هنرمند

اشتراک‌گذاری آهنگ

هر هنرمند با دستور زیر می‌تواند یک آهنگ جدید به سامانه اضافه کند.

- در هنگام اشتراک‌گذاری هر آهنگ جدید یک شناسه یکتا به این آهنگ اختصاص داده می‌شود. این شناسه‌ها از یک شروع شده و به ازای هر آهنگ جدیدی که در سامانه به اشتراک گذاشته می‌شود یک واحد افزایش می‌یابد.
- تگ‌های آهنگ به صورت رشته‌ای متشکل از کلمات که با کاراکتر ";" از هم جدا شده‌اند داده می‌شوند.
- قالب Duration آهنگ به صورت HH:MM:SS خواهد بود. همچنین ارقام خالی با صفر پر می‌شوند یعنی نمایش آهنگی با زمان 4:32 به شکل 00:04:32 می‌باشد.

ورودی

```
POST music ? title <{title}> path <{path}> year <{year}> album
<{album}> tags <{tags}> duration <{duration}>
```

خروجی

```
OK | Permission Denied | Bad Request
```

نمونه ورودی

```
POST music ? title <How I Got Home> path <~/How_I_Got_Home.mp3>
year <2023> album <How I Got Home> tags
<instrumental;classic;chill> duration <00:04:12>
```

نمونه خروجی

```
OK
```

حذف یک آهنگ

هر هنرمند با استفاده از این دستور و با وارد کردن شناسه آهنگ اشتراک‌گذاشته‌شده توسط خودش می‌تواند آن آهنگ را از فهرست آهنگ‌های موجود در سیستم حذف کند.

- در صورت حذف شدن یک آهنگ توسط هنرمند به اشتراک گذارنده آن، این آهنگ باید از پلی‌لیست‌هایی که به آن‌ها اضافه شده است و همچنین لیست آهنگ‌های هنرمند هم حذف شود.
- شناسه آهنگ حذف‌شده دیگر توسط آهنگ‌های جدید قابل استفاده نخواهد بود و آخرین شناسه موجود در سیستم نیز تغییری نخواهد کرد؛ برای مثال اگر آهنگ با شناسه ۴ از سیستم حذف شود و

در حال حاضر آخرین آهنگ اضافه شده به سیستم دارای شناسه ۲۰ باشد، شناسه آهنگ‌های ۵ تا ۲۰ تغییری نکرده و در صورت اضافه شدن آهنگ جدید، شناسه آن برابر ۲۱ خواهد بود.

- در صورتی که یک کاربر درخواستی کند که در آن از شناسه آهنگ حذف شده استفاده شود، رفتار دستور مانند شرایطی خواهد بود که این شناسه وجود ندارد؛ مثلا در دستور «اضافه کردن آهنگ به پلی لیست» با پاسخ Not Found مواجه خواهد شد.
- در صورتی که شناسه آهنگ در سامانه موجود نباشد، با پاسخ Not Found مواجه خواهیم شد.
- همچنین اگر هنرمند برای حذف کردن شناسه آهنگی را وارد کند که متعلق به خودش نباشد پیام Permission Denied دریافت می‌کند.

ورودی
`DELETE music ? id <{id}>`
خروجی
`OK | Not Found | Bad Request | Permission Denied`
نمونه ورودی
`DELETE music ? id <1>`
نمونه خروجی
`OK`

مشاهده آهنگ‌های هنرمند

هر هنرمند با این دستور می‌تواند لیست آهنگ‌هایش را ببیند.

- اطلاعات خروجی شامل شناسه، نام، سال انتشار، آلبوم، تگ‌ها و مدت زمان آهنگ است. برای آهنگ‌هایی که بیشتر از یک تگ وجود داشت، خروجی با ";" از هم جدا شده.
- قالب Duration آهنگ به صورت HH:MM:SS خواهد بود. همچنین ارقام خالی با صفر پر می‌شوند یعنی نمایش آهنگی با زمان 4:32 به شکل 00:04:32 می‌باشد.
- خروجی بر اساس شناسه آهنگ‌ها، به صورت صعودی مرتب شده است.
- در صورتی که آهنگی نبود پاسخ Empty نشان داده شود.

ورودی

```
GET registered_musics ?
```

خروجی

```
ID, Name, Artist, Year, Album, Tags, Duration

{id}, {name}, {artist}, {year}, {album}, {tags}, {duration}
… | Bad Request | Permission Denied | Empty
```

نمونه ورودی

```
GET registered_musics ?
```

نمونه خروجی

```
ID, Name, Artist, Year, Album, Tags, Duration

4, Zendegi, Hayedeh, 1972, Zendegi, Persian;Nostalgic, 5:15

8, Soghati, Hayedeh, 1969, Kharabati, Persian, 00:04:53
```

نکات و نحوهٔ تحویل

- تمام فایل‌های خود را در قالب یک پرونده‌ی زیپ با نام A7-<SID>.zip در صفحهٔ Elearn درس بارگذاری کنید که SID شمارهٔ دانشجویی شماست؛ برای مثال اگر شماره‌ی دانشجویی شما ۸۱۰۱۰۰۰۰۰ است، نام پروندهٔ شما باید A7-810100000.zip باشد.
    - برای مثال، نمونه فایل مورد قبول در زیر آمده است:

A7-810100000.zip

├── main.cpp

├── makefile

└── ...

- 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.
- برنامهٔ 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.
- دقت کنید که پرونده زیپ آپلودی شما باید پس از Unzip شدن شامل پرونده‌های پروژه شما (از جمله Makefile) باشد و از زیپ کردن پوشه‌ای که داخل آن فایل‌های پروژه‌تان قرار دارد خودداری فرمایید.
- برنامهٔ شما باید در سیستم عامل لینوکس و با مترجم g++ با استاندارد c++20 ترجمه و در زمان معقول برای ورودی‌های آزمون اجرا شود.
- 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.
- درستی برنامهٔ 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 sputify.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.
- 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. لطفاً تمرین را خودتان انجام دهید. در صورت کشف تقلب مطابق سیاست درس با آن برخورد خواهد شد.

نمرات

- تمیزی کد
  - رعایت کردن نام‌گذاری صحیح و انسجام[3]
  - عدم وجود کد تکراری
  - رعایت دندانه‌گذاری[4]
  - عدم استفاده از متغیرهای گلوبال
  - استفاده **صحیح** از متغیرهای ثابت[5] 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

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

---

[3] Consistency

[4] Indentation

[5] Constant