به نام خدا

تمرین http برنامه نویس وب (مهندسی اینترنت)

بهار ۹۹-۰۰

نام تمرین: HTTP Client with CLI

شما در این تمرین با استفاده از زبان برنامه نویسی مورد نظر خودتان، باید یک http client بنویسید و با استفاده از تعدادی آرگومان ورودی که برخی اجباری و برخی اختیاری هستند، می‌توان با رابط کاربری کار کرد..

اولین آرگومان ورودی به برنامه همیشه باید آدرس درخواست ما باشد (URL). آدرس درخواست باید حتماً اعتبار سنجی شود ولی برای اعتبار سنجی نیازی به چک کردن وجود همچین وب سروری نیست، فقط کافیست که قالب یک URL معتبر را در داشته باشد. دقت کنید در این ورودی، کاربر می‌تواند به طور کلی یک آدرس بزرگ شامل کاراکتر های خاص را داخل یک جفت " " وارد کند. برنامه نیز باید بتواند این موارد را مشابه بقیه آدرس های ساده تر هندل کند (چون بعضاً در برخی از آدرس ها کاراکتر هایی وجود دارد که برای زبان کنسولی که داخل آنها نوشته می‌شوند معانی خاصی دارند، با استفاده از یک جفت " " می توانیم با آنها به صورت یک رشته بزرگتر برخورد کنیم). در این قسمت اگر اعتبارسنجی آدرس ورودی با خطا مواجه شد، برنامه باید خطا دهد و خارج شود.

آرگومان بعدی --method و یا -M است که برای تعیین متد به کار می‌رود. برنامه باید از متدهای GET، POST، PATCH، DELETE و PUT پشتیبانی کند. هرچند درصورت نبود این آرگومان، متد پیشفرض GET تلقی می شود. اگر متدی غیر از این موارد برای برنامه تنظیم شد، برنامه باید خطا دهد و خارج شود.

آرگومان بعدی -H و یا --headers هست. با استفاده از این آرگومان شما می‌توانید برای درخواست خود، تعدادی header http (هدر) مشخص کنید. این آرگومان می‌تواند بارها تکرار شود و در نهایت تمام آنها باید با یکدیگر ترکیب شوند. به مثال های زیر دقت کنید:

```
foo.com –H “key1:value1,key2:value2” –H “key3:value3”
foo.com –H “key1:value1,key2:value2” –H “key3:value3,key4:value4”
```

برای جدا کردن جفت های key:value می توانید مانند مثال بالا، از یک جداکننده مثل , استفاده کنید. توجه داشته باشید اگر از این جدا کننده استفاده می کنید، طبیعتاً دیگر نباید از این کاراکتر به عنوان بخشی از value برای یک هدر استفاده کنید. برنامه باید بتواند این ورودی را بررسی و از آنها لیست هدرهای نهایی را استخراج کند و در درخواست اضافه کند. در این ورودی ها اگر کاربر یک یا چند هدر تکراری وارد کرد، باید آن مقداری که که دیرتر وارد کرده، جایگزین مورد قدیمی شود، در این حالت برنامه باید یک پیام اخطار (و نه خطا) نیز چاپ کند مبنی بر اینکه کاربر هدر تکراری وارد کرده و برنامه هدر جدیدتر را اولویت می دهد. تمامی key ها در هدر ها نیز باید در نهایت به صورت lowercase تنظیم و ارسال شوند.

مشابه بخش هدر ها، برنامه باید بتواند از کوئری پارامتر ها نیز پشتیبانی کند. آرگومان این قابلیت، -Q و –queries است. تمام موارد گفته شده در بالا باید برای کوئری پارامتر ها هم صدق کند، برای جدا کردن کوئری پارامتر هایی که می‌خواهید بیش از یک کوئری وارد کنید، بهتر است از کاراکتر & استفاده کنید و همچنین جدا کننده key و value نیز کاراکتر = باشد که با خود موضوع همخوانی داشته باشد.

برای نوشتن body برای درخواست، باید از آرگومان -D و یا --data استفاده کنید. این آرگومان به صورت پیشفرض، هدر Content-Type را باید application/x-www-form-urlencoded در نظر بگیرد. بدیهیست که در این صورت برنامه انتظار دارد ورودی آرگومان در قالب مقادیرkey1=value1&key2=value2 و ... باشد. برنامه لازم نیست درستی این قالب را چک کند، بلکه باید دقیقاً همان چیزی که کاربر وارد کرده را بدون تغییر به عنوان متن اصلی استفاده کند. در قسمت امتیازی کافیست اگر ورودی دیتا در ساختار مورد نظر نبود، یک اخطار (و نه خطا) به کاربر نمایش دهد و اخطار دهد که ورودی ای که کاربر مشخص کرده در فرمت پیشفرض x-www-form-urlencoded نیست. برنامه باید همان ورودی را (چه اشتباه و چه صحیح) که کاربر مشخص کرده به عنوان body برای درخواست تنظیم کند.

برنامه باید این قابلیت را داشته باشد که از فرمت داده json با استفاده از دستور --json پشتیبانی کند. به این صورت که برای این کار، هدر Content-Type باید application/json تنظیم شود و داده وارد شده باید در درخواست به عنوان body نوشته شود. اینجا نیز باید نکته بالا رعایت شود، یعنی حالتی که کاربر داده ای وارد می کند که در ساختار استاندارد json نیست، برنامه نباید برای خودش آنرا تصحیح یا تغییر دهد، فقط باید آنرا در درخواست بنویسد و content-type مورد نظر را تنظیم کند. حالت امتیازی آن است که برنامه همانند قسمت قبل صرفاً یک اخطار عدم تطابق داده با ساختار json نیز به کاربر نشان دهد.

توجه شود برنامه باید به هدرهایی که از طریق آرگومان H– و یا headers– تنظیم می‌شوند نسبت به هدرهایی که به صورت پیشفرض توسط data و json و یا هر آرگومان دیگری تنظیم می‌شوند، اولویت دهد. به مثال های زیر دقت کنید. تمام این مثال ها حالت های قابل قبول هستند و هیچ کدام نباید باعث ایجاد "خطا" در برنامه شود.

foo.com –H “content-type:application/json” --data “{\”name\”:\”hadi\”,\”last\”: \”taba\”}” –M POST

در اینجا یک داده با ساختار json در آرگومان data داده ایم، برنامه اصلاً نباید سعی کند که "مفهوم" درخواست را بفهمد، برنامه فقط باید با توجه به ورودی ها، درخواست را تنظیم کند و ارسال کند. در اینجا ابتدا به صورت پیشفرض هدر x-www-form-urlencoded تنظیم می‌شود و سپس توسط کاربر در آرگومان H–، override شده و مقدارش عوض می‌شود. فقط در حالتی که قسمت امتیازی را پیاده سازی کرده اید، برنامه باید یک "اخطار" به منظور یادآوری به کاربر نشان دهد که بگوید داده ای که وارد شده در ساختار x-www-form-urlencoded نیست. در واقع دستور بالا، دقیقاً باید عملکردی شبیه دستور زیر داشته باشد:

foo.com --json “{\”name\”:\”hadi\”,\”last\”: \”taba\”}” -M POST

مشابه با توضیحات بالا، دو دستور زیر نیز از نظر مفهومی عیناً مثل هم هستند و فقط در حالت دوم، باید یک اخطار عدم تطابق ساختار داده با json به عنوان یک یادآوری و یا هشدار به کاربر نشان داده شود.

foo.com --data “name=hadi&last=taba” –M POST

foo.com -H “content-type:application/x-www-form-urlencoded” --json “name=hadi&last=taba” -M POST

آرگومان امتیازی: برنامه باید با استفاده از آرگومان file-- آدرس یک فایل را بگیرد و آن را به عنوان body درخواست قرار دهد. آدرسی که در برنامه وارد می کنید می‌تواند absolute و یا relative باشد. توجه کنید در این حالت در صورت انتخاب نشدن هدر content-type توسط کاربر، باید این هدر content-type به صورت پیشفرض application/octet-stream بگیرد. منتها دقت شود که در نهایت هدر های مشخص شده کاربر توسط آرگومان های H- و headers-- بر همه اولویت دارند. در این آرگومان اگر فایلی در آدرس مشخص شده کاربر وجود نداشت، باید برنامه خطا دهد و خارج شود.

برنامه باید برای درخواست ها یک میزان timeout داشته باشد. این مقدار به صورت پیشفرض باید بینهایت ثبت شود و برنامه خودش درخواستی را نبندد. با استفاده از آرگومان timeout-- می‌توان یک مقدار عددی بر حسب ثانیه وارد کرد. برنامه باید پس از ارسال درخواست، به اندازه این مقدار صبر کند و اگر تا قبل از این زمان، پاسخ درخواست ما بیاید، داده را بگیرد و در غیر این صورت، درخواست را ببندد و پیام مناسب را به کاربر نشان دهد. توجه کنید این تایمر فقط از لحظه شروع ارسال تا لحظه ای که اولین پاسخ از سرور مقصد می رسد فعال است، یعنی هنگام دریافت پاسخ، دیگر این تایمر موضوعیت ندارد و باید ندید گرفته شود و اینطوری نباشد که هنگام دریافت یک فایل حجیم به عنوان پاسخ درخواست، برنامه بگوید زمان درخواست تمام شده و پیام مبنی بر تایم-اوت شدن درخواست را نشان دهد.

قابلیت لودینگ درخواست در برنامه یک قابلیت امتیازی است. پیاده سازی لودینگ در درخواست های http (چون اطلاعی از زمان دقیق رسیدن پاسخ نداریم)، ترفند های ساده ای دارد که از آنها استفاده می‌شود و جستجوی این موارد بر عهده خودتان است. همانطور نحوه نمایش لودینگ (درصد، یک نوار پر شونده و ...) نیز بر عهده خودتان است. لودینگ از لحظه ای شروع می شود که برنامه اقدام به ارسال درخواست می کند و لحظه که پاسخ درخواست به طور کامل دریافت شد، برنامه باید لودینگ را به حالت نهایی (۱۰۰ درصد، پر و ...) برساند. توجه کنید اگر این مورد را پیاده سازی می‌کنید، باید با توجه به تحلیل خودتان این مورد را با آرگومان timeout ترکیب کنید.

در پاسخ درخواست هایتان، باید حتما method، status code و status massage و تمام header های پاسخ را نشان دهید. نمایش صحیح و خوانای این موارد اهمیت دارد و نباید همه موارد را پشت سر هم در کنسول بریزید، و حداقل با یک سطر بندی مناسب و... آنهارا چاپ کنید، همچنین باید تمام body پاسخ را در

کنسول چاپ کنید. خوانایی این مورد در کنسول اهمیتی ندارد. مثلا اگر در جواب درخواست، شما یک فایل آمده، فایل را به همان صورت در کنسول چاپ کنید (dump) و عدم نمایش صحیح آن در کنسول، ایرادی ندارد. توجه داشته باشید که تمام پاسخ های http با هر status code حتی بالا ۴۰۰، می‌تواند body داشته باشد، برنامه شما باید بتواند body هر نوع درخواست را بخواند.

حالت امتیازی در این قسمت، توانایی تشخیص نوع پاسخ از روی هدر های پاسخ و انجام دادن عملیات مناسب است. این مورد باید با حالت بالا ترکیب شود، مثلا اگر یک فایل pdf دریافت کردید و متوجه شدید که یک فایل pdf است، می توانید به تحلیل خود یک نام برای آن انتخاب کرده و آنرا در یک مسیر خاص ذخیره کنید و آن را در کنسول dump نکنید. این تشخیص نوع پاسخ، باید برای فایل های pdf، عکس ها (دو فرمت jpg, png) و فیلم ها (حداقل یک فرمت) وجود داشته باشد. در این قسمت اگر سرور به طور دقیق مشخص نکرده بود که فایل مورد نظر از چه نوعی است، طبیعتاً باید همان حالت اول را اجرا کنید و آن را در کنسول dump کنید.

دقت کنید برنامه باید با دستورات شما اجرا شود، درخواست را ارسال کند، به سرانجام برسد و خارج شود و کنسول شما آزاد شود.برنامه **نباید** پس از اجرا به صورت prompt و مرحله به مرحله از شما ورودی بخواهد و بعد عملکرد ها را انجام دهد. عملکرد دراین مورد باید مشابه ابزار curl باشد.

در پیاده سازی این تمرین، اولاً می‌توانید از هر زبان برنامه نویسی استفاده کنید. برای parse کردن آرگومان های ورودی، می‌توانید از کتابخانه های آماده استفاده کنید. هدف این تمرین ایجاد پیچیدگی در این بخش نیست. برای ارسال و دریافت درخواست ها و پاسخ های http، از ماژول های از پیش تعبیه شده در زبان های برنامه نویسی استفاده کنید. در این قسمت نباید کتابخانه خیلی سطح بالا و یا خیلی سطح پایین (در سطح Socket) استفاده کنید. به عنوان مثال می‌توانید در جاوااسکریپت با استفاده از ماژول http، در پایتون با استفاده از requests و در جاوا با استفاده از HttpUrlConnection و یا HttpClient تمرین را انجام دهید. توجه کنید این موارد صرفاً مثال هستند و هیچ اجبار و یا اولویتی ندارند و استفاده از ابزار هایی در سطوح مشابه این ابزار ها موردی ندارد. طبیعتاً ممکن است برخی از این ابزار ها نقاط ضعف و قوتی داشته باشند که باید خودتان با دانشی که در این ترم ها کسب کرده اید اقدام به اصلاح آنها کنید.

نمونه مثال هایی از نحوه ورودی و خروجی برنامه در زیر آمده است. برای تست کارایی برنامه خود می‌توانید اولاً از برنامه curl برای مقایسه، ثانیاً از برخی وبسایت های آنلاین که اطلاعات درخواست شما را نشان می‌دهند (مثل https://webhook.site/) استفاده کنید و در نهایت با ریکوئست زدن و گرفتن جواب از هر وبسروری در اینترنت که وجود دارد استفاده کنید!. توجه کنید که هندل کردن https اجباری نیست اما بسیاری از کتابخانه ها به صورت پیشفرض از این پروتوکل پشتیبانی می‌کنند و عملاً نیازی به پیاده سازی جدا ندارد.

این نکته مد نظر است که ممکن است پاسخی که شما دریافت می‌کنید، لزوماً عیناً شبیه مثال هایی که در زیر آمده نباشد، نکته مهم این است مطمئن شوید هدر هایتان، body درخواست و موارد دیگر را به درستی به سرور ارسال کرده اید و اطلاعات رسیده از سرور را به درستی نمایش می‌دهید.

فرض شده که برنامه با استفاده از کلمه main در کنسول فراخوانی می‌شود، این مورد بسته به نوع پیاده سازی شما طبیعتاً متفاوت خواهد بود و موردی ندارد. اینجا صرفاً برای مثال است.

کلمه ...REST OF RESPONSE BODY... در این متن صرفاً برای خلاصه کردن پاسخ های طولانی استفاده شده و برنامه واقعی باید تمام بادی را در کنسول بنویسد.

```
>main hadi

Error: Malformed URL.
```

```
>main http://www.google.com –M hadi

Error: Method “hadi” is not recognized. Use GET, POST, PUT, PATCH and DELETE instead.
```

```
>main http://www.google.com

HTTP/1.1 200 OK
Date: Wed, 03 Mar 2021 13:24:07 GMT
Expires: -1
Cache-Control: private, max-age=0
Content-Type: text/html; charset=ISO-8859-1
P3P: CP="This is not a P3P policy! See g.co/p3phelp for more info."
Server: gws
X-XSS-Protection: 0
X-Frame-Options: SAMEORIGIN
Set-Cookie: 1P_JAR=2021-03-03-13; expires=Fri, 02-Apr-2021 13:24:07 GMT; path=/; domain=.google.com; Secure
Set-Cookie; expires=Thu, 02-Sep-2021 13:24:07 GMT; path=/; domain=.google.com; HttpOnly
Accept-Ranges: none
Vary: Accept-Encoding
Transfer-Encoding: chunked
```

<!doctype html><html itemscope="" itemtype="http://schema.org/WebPage" lang="en-IR"><head><meta content="Search the world's information, including webpages, images, videos and more. Google has many special features to help you find exactly what you're looking for." name="description"><meta content="noodp" name="robots"><meta content="text/html; charset=UTF-8" http-equiv="Content-Type"><meta content="/images/branding/googleg/1x/googleg_standard_color_128dp.png" itemprop="image"><title>Google</title><script nonce="ShOv739Qa7oJhd8YjKJfdQ==">(function(){window.google={kEI:'940_YPbKJs-tgQb35K6gBQ',kEXPI:'0,1302440,56969
…… REST OF RESPONSE BODY….

>main http://www.google.com -M POST

HTTP/1.0 411 Length Required
Content-Type: text/html; charset=UTF-8
Referrer-Policy: no-referrer
Content-Length: 1564
Date: Wed, 03 Mar 2021 13:08:13 GMT

<!DOCTYPE html>
<html lang=en>
  <meta charset=utf-8>
  <meta name=viewport content="initial-scale=1, minimum-scale=1, width=device-width">
  <title>Error 411 (Length Required)!!1</title>
  <style>
    *{margin:0;padding:0}html,code{font:15px/22px arial,sans-serif}html{background:#fff;color:#222;padding:15px}body{margin:7% auto 0;max-width:390px;min-
…REST OF RESPONSE BODY…

>main http://www.google.com -M POST --data "hadi"

-Warning: Your provided information is not of type x-www-form-urlencoded

HTTP/1.1 405 Method Not Allowed
Allow: GET, HEAD
Date: Wed, 03 Mar 2021 13:16:52 GMT
Content-Type: text/html; charset=UTF-8
Server: gws
Content-Length: 1589
X-XSS-Protection: 0
X-Frame-Options: SAMEORIGIN

<!DOCTYPE html>
<html lang=en>
  <meta charset=utf-8>
  <meta name=viewport content="initial-scale=1, minimum-scale=1, width=device-width">
  <title>Error 405 (Method Not Allowed)!!1</title>
  <style>
    *{margin:0;padding:0}html,code{font:15px/22px arial,sans-serif}html{background:#fff;color:#222;padding:15px}body{margin:7% auto 0;max
…REST OF RESPONSE BODY…

>main https://my-json-server.typicode.com/typicode/demo/comments --method GET

HTTP/1.1 200 OK
Date: Wed, 03 Mar 2021 13:31:57 GMT
Content-Type: application/json; charset=utf-8
Content-Length: 134
Connection: keep-alive
Set-Cookie: __cfduid=*********; expires=Fri, 02-Apr-21 13:31:57 GMT; path=/; domain=.typicode.com; HttpOnly; SameSite=Lax
X-Powered-By: Express
Vary: Origin, Accept-Encoding
Access-Control-Allow-Credentials: true
Cache-Control: no-cache
Pragma: no-cache
Expires: -1
X-Content-Type-Options: nosniff

Etag: W/"86-**********"
Via: 1.1 vegur
CF-Cache-Status: DYNAMIC
cf-request-id: 0899e2***************
Expect-CT: max-age=604800, report-uri="https://report-uri.cloudflare.com/cdn-cgi/beacon/expect-ct"
Report-To: {"max_age":604800,"group":"cf-nel","endpoints":[{"url":"https:\/\/a.nel.cloudflare.com\/report?s=*****"}]}
NEL: {"report_to":"cf-nel","max_age":604800}
Server: cloudflare
CF-RAY: *****-FRA

[
 {
  "id": 1,
  "body": "some comment",
  "postId": 1
 },
 {
  "id": 2,
  "body": "some comment",
  "postId": 1
 }
]

با آروزی موفقیت و سلامتی خود و خانوادتان