مدرس: رامتین خسروی
طراحان: مهیار کریمی، پرنیان فاضل، احمدرضا سلیمی،علی زارع

مهلت تحویل: جمعه ۷ آبان ۱۴۰۰، ساعت ۲۳:۵۵

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 مسؤولیت‌ها 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 قرار گرفته مراجعه کنید.

برنامه‌ریزی گردش یک‌روزه

در این تمرین قرار است به گردشگران کمک کنید تا برنامه مطلوبی برای بازدید از مکان‌های دیدنی یک شهر داشته باشند. ذکر این نکته قابل توجه است که هر یک از مکان‌های دیدنی دارای بازه زمانی مشخصی برای بازدید گردشگران است و بازدید از هر مکان نیز مدت زمان خاص خودش را می‌طلبد. علاوه بر این، هر مکان دیدنی بر اساس نظرات گردشگران دیگر حائز یک رتبه در فهرست اماکن گردشگری شهر است که نشان می‌دهد آن مکان چقدر برای بازدید جذاب است. برنامه شما با الگوریتمی که توضیح داده خواهد شد، به گردشگران کمک می‌کند تا تعداد مکان‌های جذاب بیشتری را در یک روز بازدید نمایند.

اطلاعات لازم درباره‌ی مکان‌های دیدنی این شهر در فایلی به عنوان اطلاعات ورودی به شما داده می‌شود.

ساختار فایل ورودی

فرمت فایل ورودی CSV[2] است. در این نوع فایل داده‌ها توسط کاما (,) از یکدیگر جدا می‌شوند. برنامه‌ی شما باید در ابتدا محتویات این فایل را بخواند، اطلاعات آن را به شیوه‌ی مناسب تجزیه و در ساختمان داده مناسبی ذخیره کند.

خط اول این فایل حاوی عنوان ستون‌ها است. ترتیب این ستون‌ها ثابت نیست و عنوان آن‌ها به شکل زیر است:

[1] Top Down Design
[2] Comma Separated Values

توضیح	مثال	عنوان ستون در فایل
نام مکان دیدنی	Elgoli Park	name
ساعت باز شدن به شکل hh:mm	08:00	openingTime
ساعت بسته شدن به شکل hh:mm	16:30	closingTime
رتبه یکتا	2	rank

برای مثال این فایل می‌تواند به هر دو شکل زیر باشد:

```
name,openingTime,closingTime,rank
Azadi Tower,10:30,12:00,2
National Museum,09:00,15:30,1
```

```
closingTime,name,rank,openingTime
15:30,National Museum,1,09:00
12:00,Azadi Tower,2,10:30
```

در خواندن ورودی می‌توانید فرض کنید که مقادیر همه‌ی ستون‌ها معتبر هستند و رتبه‌ی هیچ ۲ مکانی یکسان نیست.

نحوه به دست آوردن برنامه مطلوب

برای پیدا کردن برنامه مطلوب بین مکان‌های گردشگری، در هر مرحله مکانی که الان باز است و اگر وجود نداشت، مکانی که زمان باز شدن آن از همه به زمان فعلی نزدیک‌تر است را انتخاب می‌کنیم. اگر چند مکان این خصوصیت را داشتند، مکانی که رتبه کمتری دارد را در نظر می‌گیریم. می‌خواهیم در صورت امکان، دقیقا ۱ ساعت از وقتی که به هر مکان می‌رسیم، در آن‌جا بمانیم. در صورتی که پس از رسیدن به هر مکان تا زمان بسته شدن آن، کمتر از یک ساعت باقی مانده بود، همان مقدار ممکن در آن‌جا می‌مانیم ولی اگر این زمان ممکن کمتر از ۱۵ دقیقه بود به آن مکان نمی‌رویم و مکان دیگری را انتخاب می‌کنیم. پس از بازدید از یک مکان، مقصد بعدی را به همین روش انتخاب می‌کنیم و این کار را تا جایی که دیگر مکانی باز نباشد، تکرار می‌کنیم. تضمین می‌شود که همه مکان‌های دیدنی تا قبل از ساعت ۲۴ بسته خواهند شد و هیچ مکانی پیش از ساعت ۸ باز نخواهد شد. هم‌چنین فرض کنید گردش را از زودترین زمان ممکن شروع می‌کنیم.

همچنین برای جابه‌جایی میان دو مقصد، به ۳۰ دقیقه زمان نیاز است (توجه کنید رسیدن به اولین مکان نیازی به این زمان ندارد).

برای مثال برای بازدید از مکان‌های زیر به این شکل عمل می‌کنیم:

نام	ساعت باز شدن	ساعت بسته شدن	رتبه یکتا
برج آزادی	۰۸:۰۰	۱۲:۰۰	۲
موزه ملی	۰۹:۱۰	۱۰:۰۰	۱
کاخ گلستان	۰۹:۱۰	۱۰:۴۰	۳

در ابتدا برج آزادی زودتر از همه باز می‌شود پس آن‌جا را به عنوان مقصد اول انتخاب می‌کنیم. از ساعت ۰۸:۰۰ تا ۰۹:۰۰ در برج آزادی می‌مانیم. برای مقصد بعدی دو انتخاب داریم که هر دو ۹:۱۰ باز می‌شوند ولی چون رتبه موزه ملی بهتر است پس آن‌جا را انتخاب می‌کنیم و با احتساب ۳۰ دقیقه جابه‌جایی، از ۹:۳۰ تا ۱۰:۰۰ در این مکان می‌مانیم. برای مقصد بعدی یک انتخاب کاخ گلستان را داریم که در حال حاضر باز است ولی برای بازدید ساعت ۱۰:۳۰ به آن‌جا می‌رسیم و تا زمان بسته شدن ۱۰ دقیقه فاصله است. چون این زمان کمتر از ۱۵ دقیقه است در نتیجه به این مکان نمی‌رویم.

شکل خروجی

در خروجی استاندارد باید مکان‌های دیده شده را به ترتیب زمان بازدید آن‌ها به شکل زیر چاپ کنید:

```
Location <name>
Visit from <hh:mm> until <hh:mm>
---
Location <name>
Visit from <hh:mm> until <hh:mm>
---
// ...
```

ورودی و خروجی نمونه

فایل csv:

```
name,openingTime,closingTime,rank
Golestan Palace,09:10,10:40,3
National Museum,09:10,10:00,1
Azadi Tower,08:00,12:00,2
```

خروجی:

```
Location Azadi Tower
Visit from 08:00 until 09:00
---
Location National Museum
Visit from 09:30 until 10:00
---
```

نحوه دریافت فایل ورودی

آدرس فایل ورودی در ابتدا توسط آرگومان‌های خط فرمان به برنامه داده می‌شود. برای آشنایی با آرگومان‌های خط فرمان به پیوست مراجعه کنید.

برای مثال اگر نام فایل اجرایی شما a.out باشد، برنامه با دستور زیر اجرا خواهد شد:

```
./a.out ./schedule.csv
```

نحوهٔ تحویل

- کد خود را در قالب یک فایل با نام A3-SID.cpp در صفحهٔ eLearn درس بارگذاری کنید که SID شمارهٔ دانشجویی شماست؛ برای مثال اگر شماره‌ی دانشجویی شما ۸۱۰۱۹۹۹۹۹ باشد، نام پروندهٔ شما باید A3-810199999.cpp باشد که شامل کد شما است.

- برنامهٔ شما باید در سیستم عامل لینوکس و با مترجم g++ با استاندارد c++11 ترجمه و در زمان معقول برای ورودی های آزمون اجرا شود.

- تمیزی کد، ذخیره کردن اطلاعات در ساختارهای مناسب، شکستن مرحله‌به‌مرحلهٔ مسئله و طراحی مناسب، در کنار تولید خروجی دقیق و درست، بخش مهمی از نمرهٔ شما را تعیین خواهد کرد.

- درستی برنامهٔ شما از طریق آزمون‌های خودکار سنجیده می‌شود؛ بنابراین پیشنهاد می‌شود که با استفاده از ابزارهایی مانند diff خروجی برنامه خود را با خروجی‌هایی که در اختیارتان قرار داده شده است مطابقت دهید.

- هدف این تمرین یادگیری شماست. لطفاً تمرین را خودتان انجام دهید. در صورت کشف تقلب مطابق قوانین درس با آن برخورد خواهد شد.

نکات پایانی

- در این تمرین اجازه استفاده از شیءگرایی و makefile را ندارید.

- تضمین می‌شود که نام فایل از طریق آرگومان‌های خط فرمان به شما داده می‌شود و همچنین این فایل به فرمت گفته شده وجود دارد.
- با توجه به اینکه تجزیه پرونده‌های CSV بخشی از تمرین است، استفاده از کتابخانه‌های موجود برای تجزیه فایل csv قابل قبول نیست.

پیوست

آرگومان‌های خط فرمان:

آرگومان‌های خط فرمان آرگومان‌هایی هستند که سیستم‌عامل در زمان اجرای برنامه آن‌ها را به برنامه انتقال می‌دهد. برنامه می‌تواند آن‌ها را نادیده بگیرد و یا از آن‌ها استفاده کند.

برای استفاده از این آرگومان‌ها، تابع main باید به صورت زیر نوشته شود:

```
int main(int argc, char* argv[])
```

دو آرگومان تابع را می‌توان برای دسترسی به آرگومان‌های خط فرمان استفاده کرد:

- argc

  عدد صحیح؛ تعداد آرگومان‌های خط فرمان داده شده به برنامه
  این مقدار حداقل برابر با یک است؛ زیرا دستور اجرای برنامه (نام پرونده اجرایی) حتماً در زمان اجرای برنامه مورد استفاده قرار می‌گیرد و همواره به‌عنوان آرگومان‌های خط فرمان شماره صفر به برنامه داده می‌شود.

- argv

  آرایه‌ای از رشته‌های مدل زبان C؛ آرگومان‌های خط فرمان داده شده به برنامه

به عنوان یک مثال ساده برنامه زیر را در نظر بگیرید:

```cpp
#include <iostream>

int main(int argc, char *argv[])
{
   std::cout << "There are " << argc << " arguments:" << std::endl;

   // Loop through each argument and print its number and value
   for (int count=0; count < argc; ++count)
      std::cout << count << " " << argv[count] << std::endl;

   return 0;
}
```

اگر برنامه به شکل

./a.out myFile.txt 100

اجرا شود، خروجی زیر تولید می‌شود:

```
There are 3 arguments:
0 ./a.out
1 myFile.txt
2 100
```

برای آشنایی بیشتر با نحوه کار آرگومان‌های خط فرمان می‌توانید به این لینک مراجعه کنید.