مدرس: رامتین خسروی
طراحان: علی زارع، کیانوش عرشی، پاشا براهیمی، کیمیا محمدطاهری

مهلت تحویل: جمعه ۲۷ اسفند 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 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 قرار گرفته مراجعه کنید.

همایش‌های بین‌المللی

هر ساله تعدادی همایش بین‌المللی در روز جهانی گردشگری برگزار می‌شود. در این همایش‌ها افراد با ملیت‌ها و زبان‌های متفاوت حضور دارند، بنابراین نیاز به تعدادی مترجم داریم. در این تمرین شما باید برای هر کدام از همایش‌ها، مترجم‌های لازم را مشخص کنید. دقت کنید که مترجم‌ها هر کدام زبان‌های خاصی را بلد هستند و همچنین در زمان‌های مشخصی امکان شرکت در همایش‌ها را دارند. برنامه شما با الگوریتمی که توضیح داده خواهد شد، زمان‌بندی مناسب برای مترجم‌ها را انجام می‌دهد. توجه کنید که ممکن است زبانی بدون مترجم باقی بماند که این مورد باید در خروجی ذکر شود.

اطلاعات لازم درباره‌ی مترجم‌ها و همایش‌ها در فایلی به عنوان اطلاعات ورودی به شما داده می‌شود.

مترجم

برای هر مترجم نام او، زبان‌هایی که بلد است و بازه زمانی که آزاد است به عنوان ورودی داده می‌شود.

همایش

برای هر همایش نام، بازه زمانی برگزاری و زبان‌های نیازمند مترجم در ورودی داده می‌شود.

[1] Top-Down Design

نحوه بدست آوردن زمان‌بندی مناسب

برنامه زمان‌بندی به این صورت کار میکند که همایش‌ها را به ترتیبی که در ورودی داده شده بررسی می‌کند و مترجم های مناسب را اختصاص می‌دهد. دقت کنید که در بررسی یک همایش، اول برای زبانی مترجم پیدا می‌کنیم که کمترین تعداد مترجم را دارد. همچنین اگر تعداد مترجم‌ها برای دو زبان برابر بود، زبانی اولویت دارد که در ورودی زودتر آمده است. واضح است که برای هر زبان، مترجمی انتخاب می‌شود که در کل بازه زمانی برگزاری همایش آزاد باشد. اگر چند مترجم در زمان برگزاری همایش آزاد بودند، مترجمی انتخاب می‌شود که تعداد زبان کمتری بلد است. همچنین اگر چند مترجم تعداد زبان برابری بلد بودند، مترجمی که نام او از لحاظ ترتیب الفبایی کوچکتر باشد، در اولویت است. اگر یک مترجم برای یک همایش انتخاب شود، بازه زمانی برگزاری همایش از زمان آزاد او حذف می‌شود و در آن زمان نمی‌تواند برای همایش دیگری انتخاب شود. دقت کنید که مترجم انتخاب شده برای همایش فقط می‌تواند مسئولیت ترجمه **یک زبان** در آن همایش را به عهده بگیرد. همچنین ممکن است برای یک یا چند زبان مترجمی پیدا نشود.

قالب فایل ورودی

اطلاعات مترجم‌ها و همایش‌ها در یک فایل وجود دارد که آدرس آن از طریق آرگومان خط فرمان به برنامه داده می‌شود.

فایل ورودی شامل دو بخش است که بخش اول مربوط به مترجم‌ها و بخش دوم مربوط به همایش‌ها است.

در خط اول تعداد مترجم‌ها (n) داده می‌شود و در n خط بعدی اطلاعات هر مترجم با قالب زیر وارد می‌شود:

```
<translator_name> <start_time> <end_time> <languages>
```

دقت کنید که start_time و end_time به فرمت hh:mm هستند و نشان‌دهنده ساعت آغاز و پایان وقت آزاد مترجم هستند.

همچنین languages زبان‌هایی هستند که مترجم بلد است و با فاصله[2] از هم جدا شده‌اند.

بعد از اتمام اطلاعات مترجم‌ها، تعداد همایش‌ها (k) داده می‌شود. در k خط بعدی نیز اطلاعات هر همایش با قالب زیر وارد می‌شود:

```
<event_name> <start_time> <end_time> <languages>
```

start_time و end_time به فرمت hh:mm هستند.

همچنین languages زبان‌هایی هستند که مترجم لازم دارند و با فاصله از هم جدا شده‌اند.

[2] Space

قالب خروجی

برای هر همایش ابتدا در یک خط نام آن همایش و در خط‌های بعدی در هر خط یکی از زبان‌های موجود در همایش و در کنار آن، نام مترجم مربوط به آن زبان چاپ می‌شود. اگر برای یک زبان مترجمی پیدا نشده بود، به جای نام مترجم عبارت Not Found چاپ می‌شود. دقت کنید که همایش‌ها و زبان‌های یک همایش به ترتیبی که در ورودی داده شده‌اند در خروجی چاپ می‌شوند.

```
<event1_name>
<language1>: <translator>
<language2>: <translator>
…
<event2_name>
<language1>: <translator>
<language2>: <translator>
…
```

ورودی و خروجی نمونه

توجه کنید که برای اطمینان کامل از عملکرد برنامه‌تان لازم است خودتان آزمون‌های بیشتری طراحی کنید.

نمونه ۱:

ورودی	خروجی
2 Hamid 08:00 17:30 Persian English Spanish Amin 07:30 15:00 French Arabic 2 Event1 09:00 10:00 French Event2 12:00 13:00 Arabic Spanish	Event1 French: Amin Event2 Arabic: Amin Spanish: Hamid

ابتدا باید برای Event1 مترجم زبان French پیدا کنیم که در بین مترجم‌ها فقط Amin خصوصیت لازم را دارد پس انتخاب می‌شود. مترجم‌های همایش Event2 هم به همین شکل انتخاب می‌شوند.

نمونه ۲:

ورودی	خروجی
2 Hamid 08:00 17:30 Arabic Amin 10:00 15:00 French Arabic 1 NationalEvent 12:00 13:00 Arabic	NationalEvent Arabic: Hamid

باید برای NationalEvent مترجم Arabic پیدا کنیم. در بین مترجم‌ها دو نفر این زبان را بلدند و در زمان همایش آزاد هستند، ولی Hamid تعداد زبان کمتری بلد است پس انتخاب می‌شود.

نمونه ۳:

ورودی	خروجی
2 Hamid 08:00 17:30 Arabic Persian Amin 10:00 15:00 French Arabic 3 Event1 12:00 14:00 Arabic Event2 12:00 14:00 Arabic Event3 13:00 15:00 Persian Spanish	Event1 Arabic: Amin Event2 Arabic: Hamid Event3 Persian: Not Found Spanish: Not Found

برای Event1 مترجم زبان Arabic نیاز است. هر دو مترجم در این زمان آزاد هستند و زبان را بلدند. چون تعداد زبان مساوی بلدند در نتیجه Amin که اسمش در الفبا زودتر می‌آید انتخاب می‌شود.

در Event2 هم زبان Arabic لازم است. چون Amin در آن بازه زمانی آزاد نیست فقط Hamid می‌ماند و انتخاب می‌شود.

برای Event3 حمید زبان Persian را بلد است ولی در زمان همایش آزاد نیست پس کسی انتخاب نمی‌شود.

برای زبان Spanish هم مترجم نداریم و در نتیجه برای این زبان هم کسی انتخاب نمی‌شود.

نمونه ۴:

ورودی	خروجی
2 Hamid 08:00 17:30 Arabic English Spanish Amin 07:30 15:00 French Arabic Spanish 1 TehranTourism 09:00 10:00 Spanish French	TehranTourism Spanish: Hamid French: Amin

در همایش TehranTourism ابتدا برای زبان French مترجم پیدا می‌کنیم چون تعداد مترجم کمتری دارد (زبان Spanish دو مترجم دارد و زبان French یک مترجم). برای این زبان Amin خصوصیات لازم را دارد و انتخاب می‌شود. سپس باید برای Spanish مترجم پیدا کنیم. هر دو مترجم زبان را بلدند ولی Amin وقت آزاد ندارد پس Hamid انتخاب شده است.

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 ./input.txt
```

نحوه خواندن فایل ورودی

برای خواندن محتوای فایل ورودی می‌توانید از کتابخانه fstream استفاده کنید. اطلاعات بیشتر درباره این کتابخانه را در این لینک می‌توانید مطالعه کنید.

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 ۸۱۰۱۰۰۰۰۰ باشد، نام پروندهٔ شما باید A3-810100000.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.

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.