استاد: **رامتین خسروی**
طراح: **رامتین خسروی**، امین ستایش

مهلت تحویل: جمعه ۶ اسفند ۱۴۰۰، ساعت ۲۳:۵۵

## مقدمه

هدف از این تمرین آشنایی شما با ورودی و خروجی و کار با رشته‌ها و بردارها در زبان C++ به عنوان مهارت‌های پایه‌ای برنامه‌نویسی در این زبان است. این مهارت‌ها پیاده‌سازی پروژه‌های بزرگتر را امکان‌پذیر می‌کنند. در این تمرین شما یک برنامه‌ی تنظیم کارهای روزانه را پیاده‌سازی می‌کنید.

علاوه بر درستی برنامه، تمیزی کد و جدا کردن برنامه به توابع مختلف اهمیت زیادی دارند. سعی کنید که قبل از انجام تمرین حتما ویدیوهای بخش تمیزی کد موجود در صفحه‌ی درس را مشاهده کنید.

## تنظیم کارهای روزانه

در این تمرین شما به پیاده‌سازی یک سامانه‌ی تنظیم کارهای روزانه می‌پردازید. این سامانه قابلیت برنامه‌ریزی کارهای فرد در بین جلسات ثابت طول روز را دارد. در این سامانه ابتدا تعدادی بازه‌ی زمانی به عنوان جلسات ثابت طول روز اعلام می‌شوند و در ادامه کارهای فرد، شرایط انجام و مدت‌زمان مورد نیاز هر کدام مشخص می‌شوند. در نهایت سامانه ترتیبی برای زمان انجام کارها چاپ می‌کند.

### جلسات روزانه

در طول روز زمان‌های ثابتی به عنوان زمان جلسات در سامانه تعیین می‌شوند. این بازه‌های زمانی با یک زمان شروع و یک زمان پایان با فرمت hh:mm (برای مثال 13:00) تعیین می‌شوند. تضمین می‌شود که زمان‌های جلسات در ورودی به ترتیب صعودی زمان شروع داده خواهند شد و جلسات با یکدیگر تداخلی نخواهند داشت. البته لازم به ذکر است که یکسان بودن زمان پایان یک جلسه و زمان شروع جلسه بعدی به معنای تداخل نیست. برای مثال یک جلسه می‌تواند ساعت ۱۳:۰۰ تمام شود و دیگری ساعت ۱۳:۰۰ شروع شود.

### کارهای روزانه

پس از مشخص شدن زمان‌های ثابت جلسات، کارهای روزانه در ورودی ذکر می‌شوند. برای هر کار مدت‌زمان مورد نیاز برای اتمام آن کار و بازه‌ی زمانی آزاد برای انجام آن کار مشخص می‌شود. در بخش بعدی در رابطه با بازه‌ی زمانی انجام کار توضیح داده شده‌است. به هر کار یک عدد صحیح یکتا اختصاص داده می‌شود که در ادامه‌ی برنامه با استفاده از این عدد صحیح هر کار به صورت یکتا مشخص می‌شود. برای اختصاص عدد شناسه یکتا به کارها کافی است که به اولین کار موجود در ورودی عدد ۱ و در ادامه ۲ و ... اختصاص دهید.

## بازه‌های زمانی آزاد

همانطور که در بخش قبل ذکر شد، در ورودی برای هر کار بازه‌ی زمانی که باید کار در آن زمان انجام شود مشخص می‌شود. هر بازه‌ی پیوسته‌ای که در آن جلسه برگزار نمی‌شود به صورت یک بازه‌ی زمانی آزاد شماره‌گذاری می‌شود. همچنین این بازه‌ها به ترتیب از ۱ شماره‌گذاری می‌شوند. **دقت داشته باشید** که شروع روز ساعت ۱۲:۰۰ و پایان روز ساعت ۲۰:۰۰ محسوب می‌شود. هیچ کاری قبل و بعد از این دو عدد نمی‌تواند قرار بگیرد. برای بهتر متوجه شدن نحوه شماره‌گذاری به مثال زیر دقت کنید.


در مثال بالا زمان‌های جلسات با مستطیل‌های آبی رنگ مشخص شده‌اند و بازه‌های زمانی آزاد با اعداد سبز شماره‌گذاری شده‌اند.

## برنامه‌ریزی کارها

سامانه در نهایت کارهای مشخص‌شده را در بین بازه‌های زمانی آزاد می‌چیند به طوری که هر کار در بازه‌ی زمانی آزاد مشخص‌شده خود قرار بگیرد. همچنین **دقت داشته باشید** اگر چند کار برای انجام در یک بازه‌ی زمانی مشخص شده باشند کاری باید زودتر انجام شود که در ورودی کارها زودتر آمده است (در واقع شناسه‌ی یکتای کمتری دارد).
کارها نباید تداخل داشته باشند و بایستی از زودترین زمان ممکن چیده شوند. تضمین می‌شود که چینش با مشخصاتی که در ورودی به شما داده می‌شود ممکن است.

## ورودی و خروجی

### قالب ورودی

در چند خط اول ورودی مشخصات جلسات به صورت زمان شروع و پایان با فرمت hh:mm مشخص می‌شوند. سپس برای مشخص شدن اتمام ورودی جلسات کاراکتر # در ورودی داده می‌شود. در ادامه نیز مشخصات کارها داده می‌شود که شامل مدت زمان انجام کار به دقیقه و بازه‌ی زمانی آزادی که کار باید در آن بازه‌ی زمانی کار انجام شود است. دقت داشته باشید که ورودی در نهایت با شناسه‌ی پایان فایل[1] (معادل Ctrl+D در هنگام ورودی دادن) به پایان خواهد رسید.

### قالب خروجی

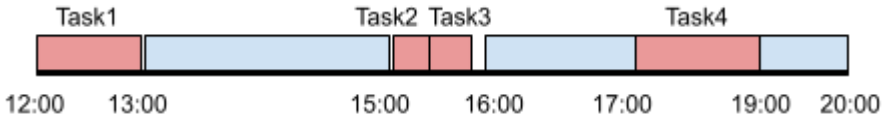
در هر خط خروجی شناسه‌ی یک کار و بازه‌ی زمانی که برای آن برنامه‌ریزی شده است مشخص می‌شود. زمان‌ها به فرمت hh:mm چاپ می‌شوند.

### ورودی و خروجی نمونه

[1] EOF

| قالب ورودی | قالب خروجی |
|---|---|
| <event_time><br><event_time><br>...<br>#<br><task_duration> <empty_range_id><br><task_duration> <empty_range_id><br>... | <task_id> <task_scheduled_time><br><task_id> <task_scheduled_time><br>…. |

| ورودی | خروجی |
|---|---|
| 13:00-15:00<br>16:00-17:00<br>19:00-20:00<br>#<br>60 1<br>30 2<br>20 2<br>120 3 | 1 12:00-13:00<br>2 15:00-15:30<br>3 15:30-15:50<br>4 17:00-19:00 |




## نحوهٔ تحویل

- کد خود را در قالب یک فایل با نام A1-SID.cpp در صفحه‌ی eLearn درس بارگذاری کنید که SID شمارهٔ دانشجویی شماست؛ برای مثال اگر شماره‌ی دانشجویی شما ۸۱۰۱۰۰۰۰۰ باشد، نام پروندهٔ کد شما باید A1-810100000.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، 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 از درستی کامل قالب خروجی برنامه خود اطمینان حاصل کنید و از دادن خروجی‌هایی که در صورت پروژه ذکر نشده‌است اجتناب کنید. برای مقایسه‌ی خروجی خود حتما از توضیحات پیوست ۱ استفاده کنید تا مشکلی در زمان تحویل در اجرای تست‌های شما پیش نیاید.

- در طول این تمرین ممکن است با مشکلاتی روبه‌رو شوید که راه حل آن‌ها را نمی‌دانید؛ در این صورت، جست‌وجوگرهایی مانند google و سایت‌هایی مانند stackoverflow و cplusplus.com ممکن است به شما کمک کنند.

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

## پیوست ۱

مقایسه خروجی برنامه با خروجی مورد انتظار با چشم شاید برای برنامه‌های کوچک که خروجی کمی تولید می‌کنند و روند اجرای کوتاهی دارند میسر باشد، اما این کار برای برنامه‌های بزرگ‌تر با مسیر اجرای پیچیده کاری دشوار است. برای این کار می‌توان از ابزارهایی که در سیستم‌عامل لینوکس در دسترس است استفاده کرد.
برای این کار با استفاده از مجموعه دستورات زیر خروجی را در فایل دلخواه ذخیره می‌کنیم.

```
g++ -std=c++11 source.cpp
./a.out < in.txt > out.txt
```

حال فرض کنیم که خروجی درست در فایل true_out.txt قرار داشته باشد، برای مقایسه‌ی این خروجی با خروجی out.txt که توسط برنامه‌ی ما تولید شد از ابزار **diff** استفاده می‌کنیم.

```
diff out.txt true_out.txt
```

اگر دو فایل یکسان باشند دستور diff هیچ خروجی‌ای تولید نمی‌کند. وگرنه، تفاوت‌های دو فایل را نمایش می‌دهد.
برای مطالعه‌ی بیشتر درباره‌ی diff می‌توانید از این لینک استفاده کنید.