مدرس: رامتین خسروی
طراحان: سامان اسلامی نظری، پاشا براهیمی

مهلت تحویل: جمعه ۴ اسفند ۱۴۰۲، ساعت ۲۳:۵۹

مقدمه

هدف از این تمرین آشنایی شما با ورودی و خروجی در کتابخانه iostream[1] و همچنین آزمون برنامه به سبک استفاده شده در درس است. در این تمرین شما یک سیستم توصیه بلیت قطار ایجاد خواهید کرد که در آن کاربران با توجه به مبدا و مقصد، می‌توانند قطارهای متناسب با نیاز خود را پیدا کنند.

سیستم توصیه بلیت قطار

هر روزه تعداد زیادی از افراد به وسیله قطار به شهرهای دیگر سفر می‌کنند. در این تمرین قصد داریم یک سیستم برای یافتن بلیت قطار جهت کمک به این افراد تهیه کنیم. این سیستم اطلاعات کاربر را دریافت کرده و تمامی بلیت‌های مناسب او را نمایش می‌دهد.

وارد کردن لیست قطار

در این مرحله، مسئول قطار لیست قطارهای روز جاری را به برنامه اضافه می‌کند. اطلاعات مربوط به هر قطار در یک خط جداگانه نوشته می‌شود. لازم به ذکر است که ترتیب خاصی برای وارد کردن لیست قطارها وجود ندارد و این قطارها ممکن است به هر ترتیبی به برنامه داده شوند. اطلاعات ورودی برای هر قطار به ترتیب شامل نام قطار، مبدا، مقصد، زمان حرکت و ظرفیت باقی‌مانده آن قطار است.

پیکربندی ورودی
`<train_name> <source> <destination> <time> <remaining_capacity>`

[1] https://www.geeksforgeeks.org/basic-input-output-c/

یافتن بلیت قطار

کاربران می‌توانند برای پیدا کردن قطار(های) مدنظر خود، مبدا، مقصد، اولین زمان ممکن برای حرکت قطار و تعداد بلیت‌های مورد نیازشان را وارد برنامه کنند تا قطارهای ممکن برایشان نمایش داده شود. نحوه ورود اطلاعات در ادامه قابل مشاهده است.

پیکربندی ورودی
`<source> <destination> <first_possible_time> <count>`

ورودی

در اولین خط به ترتیب دو عدد n و m داده می‌شود که n تعداد قطارها و m تعداد درخواست‌های یافتن قطار است. سپس در n خط بعدی در هر خط اطلاعات مربوط به یک قطار داده می‌شود. پس از آن و در m خط انتهایی، در هر خط اطلاعات مربوط به یک سفر جهت یافتن قطار داده می‌شود. لازم به ذکر است که فرمت تمامی زمان‌ها به صورت hh:mm خواهد بود و محدوده قابل قبول برای زمان، برابر با 00:00 تا 23:59 است.

پیکربندی ورودی

`<n> <m>`
`<train_1>`
`<train_2>`
`...`
`<train_n>`
`<request_1>`
`<request_2>`
`...`
`<request_m>`

خروجی

به ازای هر باری که کاربر اطلاعات خود را جهت یافتن بلیت وارد می‌کند، قطارهایی که کاربر امکان رزرو آن‌ها را دارد، نمایش داده می‌شود. قطارهای نمایش‌داده‌شده باید مبدا و مقصد برابر با درخواست کاربر داشته باشد، زمان حرکتش زودتر از زمان درخواستی کاربر نباشد و حداقل به اندازه تعداد مورد نیاز کاربر ظرفیت خالی داشته باشد. لازم به ذکر است که ترتیب نمایش قطارها باید به همان صورتی باشد که در ورودی آمده است.

در خط بعد از نمایش آخرین قطار به ازای هر درخواست، تعداد ۱۰ عدد - (خط فاصله[2]) چاپ می‌شود. تضمین می‌شود به ازای هر درخواست حداقل یک قطار وجود خواهد داشت.

پیکربندی خروجی

`<train_name_1> <time_1> <remaining_capacity_1>`
`<train_name_2> <time_2> <remaining_capacity_2>`
`...`
`<train_name_n> <time_n> <remaining_capacity_n>`
`----------`

مثال

ورودی نمونه اول

نمونه ورودی

`5 2`
`Raja_501 Tehran Shiraz 23:55 5`
`Raja_511 Tehran Shiraz 21:15 8`
`Talaeie_525 Tehran Shiraz 20:05 10`
`Raja_423 Mashhad Shiraz 17:30 23`
`Noor_431 Tehran Shiraz 16:20 12`
`Tehran Shiraz 17:00 6`
`Mashhad Shiraz 00:00 2`

خروجی نمونه اول

نمونه خروجی
`Raja_511 21:15 8`

[2] Dash

```
Talaeie_525 20:05 10
----------
Raja_423 17:30 23
----------
```

نکات و نحوهٔ تحویل

- 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.
- 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 ممکن است به شما کمک کنند.
- تحویل این تمرین در سامانه کوئرا انجام می‌شود. برای ورود به کلاس در سایت کوئرا می‌توانید از این لینک استفاده کنید. رمز ورود به کلاس APS03 است.
- 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.
- دقت کنید که این تمرین نمره‌ای ندارد اما انجام آن، **اجباری** است.