Task Manager

در این پروژه قصد داریم برنامه‌ای بنویسیم که جای Wunderlist، Todoist و Things 3 را بگیرد! کلیت قضیه حول نوشتن برنامه‌ای برای مدیریت کارها است. به این صورت که کاربر کارهای خود و زمانی که می‌خواهد به آن اختصاص بدهد را مشخص می‌کند و می‌تواند در آینده به لیست کارهایش دسترسی داشته باشد. همین‌طور می‌تواند از برنامهٔ شما بخواهد که زمان خالی و مناسبی برای کارهاش پیدا کند.

ویژگی‌ها

- برنامه باید به‌طور مداوم آمادهٔ ورودی گرفتن دستور از کاربر باشد.
- با وارد کردن دستور today، برنامه باید کارهای کل روز جاری را با فرمت مناسب به کاربر نشان بدهد(از تاریخ و ساعت سیستم استفاده کنید تا امروز را تشخیص دهید).
- با وارد کردن دستور month، برنامه باید کارهای پیش‌رو در ماه جاری را به کاربر نشان بدهد.
- با وارد کردن دستور add، برنامه باید وارد حالت ورودی گرفتن تسک بشود که حالت ورودی گرفتن به صورت زیر است:
  - اگر title is TASK_TITLE وارد شود، نام تسک را TASK_TITLE در نظر می‌گیریم.
  - اگر starts in dd-mm hh:mm وارد شود، زمان شروع انجام تسک را متناسب با آن قرار می‌دهیم.
  - اگر ends in dd-mm hh:mm وارد شود، زمان اتمام تسک را متناسب با ورودی قرار می‌دهیم.
  - اگر done وارد شد، تسک را با اطلاعاتی که از کاربر گرفته است، ذخیره کند و از حالت ورود تسک خارج شود (داده‌ها در فایل ریخته شود تا پس از باز و بسته شدن برنامه، اطلاعات پایدار بماند).
- اگر find time hh:mm وارد شود، برنامه (با درنظر گرفتن آن که انجام تسک، به اندازهٔ hh ساعت و mm دقیقه طول می‌کشد) باید بازه‌ای خالی را در ماه جاری پیدا کند که به اندازه hh:mm باشد. الویت با نزدیکترین جای خالی در آینده به زمان حال است. اگر بازه کاملا خالی پیدا نکرد باید بازه ای را پیشنهاد دهد که حداقل هم‌پوشانی را با دیگر تسک‌های شخص داشته باشد.
- با وارد کردن دستور now، برنامه باید تاریخ و زمان حال را چاپ کند.
- هر باری که برنامه باز می‌شود، باید اطلاعات کاربر و کارهای او را به صورت کامل بازگردانی کند (اطلاعات را در فایل ذخیره کنید).

- با وارد کردن دستور free dd-mm hh:mm، برنامه باید آن زمان را خالی کند (یعنی تمام کارهایی که با آن زمان تداخل دارند حذف شوند). توجه کنید که فایل نیز باید بروزرسانی شود.
- با دستور schedule dd-mmکه در آن dd یک روز است باید بزرگترین زیرمجموعه از کارهای داده شده در آن روز را که قابل انجام اند چاپ کند. زیر مجموعه قابل انجام یعنی زیرمجموعه از کارها که هیچ دو تایی از آنها تداخل نداشته باشد. اگر مساله چند جواب داشت چاپ یکی از جواب‌ها کفایت میکند. یک الگوریتم حریصانه برای این مساله در کتاب طراحی الگوریتم CLRS ارایه شده است (ابتدای فصل ۱۶) نمونه. **از این الگوریتم می‌توانید ایده بگیرید اما از هیچ منبعی کد کپی نکنید**.
- برنامه باید اطلاعات مربوط به سال جاری را داشته باشد (نیازی به نگهداری سال‌های قبل و بعد نیست).

مشخصات فنی

- فایلی که برنامه اطلاعات تسک ها را نگه میدارد، به صورت باینری نگهداری شود.
- طراحی پروژه باید کاملا شی گرا باشد. فقط مجاز هستید منوی اصلی را در داخل main پیاده کنید و بقیه کد باید در ساختاری شی گرا مرتب شده باشد. تابعی خارج از یک کلاس نمره منفی دارد (البته به غیر از main).
- تعریف کارها (tasks) به صورت کلاس باشد.
- تعریف سال‌نامه به عنوان کلاس، به شکلی که شامل ماه‌های سال جاری باشد.
- تعریف ماه به عنوان کلاس، به شکلی که شامل روزهای ماه باشد.
- هر روز شامل لیستی از کارهای مربوط به روز است. از vector برای نگه داشتن این لیست استفاده کنید.

بعد از اتمام پیاده‌سازی پروژه و تست آن، همه فایل‌ها را زیپ کرده و یک فایل زیپ آپلود کنید. تصحیح این پروژه به صورت اتوماتیک توسط کویرا وجود ندارد (به علت استفاده از فایل‌ها). این پروژه به صورت دستی توسط تدریس‌یارها تصحیح شده و نمره استاد بروزرسانی خواهد شد.

ورودی/خروجی نمونه

```
~ add
~ title is Do homework
~ starts in 02-01 15:05
~ ends in 02-01 15:55
~ done
```

```
~ today
>    (1) “Do homework” [15:05 - 15:55]
~ add
~ title is Read book
~ starts in 02-01 15:45
~ ends in 02-01 21:30
~ done
~ today
>    (1) “Do homework” [15:05 - 15:55]
>    (2) “Read book” [15:45 - 21:30]
~ schedule 02-01
>    (1) “Do homework” [15:05 - 15:55]
~ free 02-01 21:01
~ today
>    (1) “Do homework” [15:05 - 15:55]
~ now
>    02-01 14:35
~ find time 1:00
>    02-01 [15:55 - 16:55]
```