به نام خالق یکتا و به یاد یوسف زهرا

گزارش تمرین برنامه‌سازی پیشرفته

تمرین سری دوم

سیدرضا هاشمی‌راد

۱۵ اسفند ۱۳۹۷

فهرست مطالب

لینک دستیابی به فایل ها در گیت هاب

۱ سوال یک

برای حل این سوال ابتدا کلاس Map را ایجاد می نماییم. در کانستراکتور آن که هنگام صدازد شدن یک ورودی که ابعاد نقشه است را می گیرد، یک وکتور دو بعدی از نوع عدد صحیح ایجاد نموده و درایه های آن را با اعداد تصادفی پر می نماییم. (در این قسمت نقشه مورد نظر آماده است.)

همچنین برای نشان دادن مسیر عبوری در قسمت های بعدی، یک وکتور دو بعدی دیگر ایجاد نموده و آن را با علامت «ـ» پر می نماییم. با توجه به اینکه می دانیم حرکت از گوشه سمت چپ شروع می شود، خانه اول را برابر «*» قرار می دهیم. (این کار را سه بار برای سه قسمت سوال انجام می دهیم یعنی سه وکتور با این ویژگی ها را تولید می نماییم.)

برای نمایش نقشه و نمایش مسیر کار ساده است. صرفا با یک range-based for loop محتوای وکتورها را خوانده و با استفاده از کتابخانه iomanip و تابع std::setw() آن را به صورت مرتب در کنسول نمایش می دهیم. (نمایش مسیر طی شده نیز دقیقا به همین گونه است و تنها کافی است اسم وکتور هدف عوض شود.)

در ادامه تابع findRoute() را می نویسیم. به این صورت که در ابتدای فراخوانی آن، موقعیت فعلی را صفر نموده و وارد یک حلقه while می شویم که شرط برقراری آن نرسیدن به خانه نهایی یعنی راست و پایین ترین (نقطه مقصد) است. این را با دو متغیر currentX و currentY که موقعیت فعلی ما را نشان می دهند، بررسی می کنیم. داخل حلقه چند شرط را باید بررسی نماییم و مطابق با آنها حرکتی صحیح را انجام دهیم. اگر شمارنده ستون ها به انتها رسیده باشد، در جهت پایین حرکت می کنیم و اگر شمارنده سطرها (currentX) به آخر رسیده باشد، در جهت راست حرکت می کنیم. (direct = 1 یعنی راست و direct = 2 یعنی حرکت به سمت چپ.)

اگر هیچ کدام از حالت های فوق نباشد، مقدار اختلاف را در دو جهت محاسبه نموده و جهت را طوری قرار می دهیم و طوری حرکت می کنیم که عدد محاسبه شده کمتر باشد. در انتها، در finalRoute مقدار متناظر currentX و currentY را برابر «*» قرار می دهیم تا به این ترتیب مسیر عبوری در این ماتریس ثبت شود.

در ادامه برای حرکت سه جهتی هم دقیقا همین کار را انجام می دهیم با این تفاوت که اگر در سطر یا ستون انتهایی نبودیم، به جای محاسبه دو اختلاف، سه اختلاف را محاسبه نموده و به سمتی حرکت می کنیم که اختلاف کمتری را محاسبه نمودیم. (در اینجا، direct = 2 به معنی حرکت در جهت گوشه است.)

تابع showRoute() یک ورودی می گیرد و آن این است که کدام مسیر را می خواهیم نمایش دهیم. مسیری که در قسمت الف طی شده یا مسیری که در قسمت ب سوال طی کردیم. اگر ورودی ۲ باشد، مسیر دو جهته را و اگر ۳ باشد مسیر سه جهته طی شده را نمایش می دهد.

در قسمت بعدی، باید ابتدا تمام حالات ممکن را تولید نماییم. برای این منظور از کد زده شده در سوال دوم تمرین سری یک استفاده می کنیم. به این صورت که ابتدا یک رشته با طول $2(n-1)$ ایجاد نموده و تمام کاراکترهای آن را با r پر می نماییم. حال با استفاده از کد سوال دوم تمرین سری اول تمام حالت ها را به صورت باینری ایجاد می نماییم. (به جای بزرگ کردن حرف در سوال دو تمرین سری اول، حرف d را قرار می دهیم. توجه شود که تنها ترکیب هایی باید به حالات ما اضافه شوند که تعداد حرکت های راست و پایین آنها کمتر از $n-1$ هستند. لذا با یک شرط این موضوع را بررسی می نماییم.

پس از تولید کلیه حالات قابل قبول، برای هر یک از حالات طبق دستورالعمل مسیر را طی نموده، میزان اختلافات را در متغیری ذخیره نموده و آن را به یک وکتور اضافه می نماییم.

در مرحله بعد کوچکترین جمع را پیدا نموده و مسیر متناظر با آن را در finalMinRoute ذخیره می نماییم و آن را نمایش می دهیم.

۲ سوال دو

برای حل این سوال مطابق آنچه در صورت سوال آمده بود، ابتدا دو کلاس خواسته شده را با نام‌های LiBVec و LibArr ایجاد می نماییم. در کلاس اول متغیر اصلی ما از نوع وکتور و در کلاس دوم از نوع آرایه معمولی است. هر کلاس یک تابع به نام counter() دارد که سایز آرایه مورد نظر را در ورودی گرفته و آرایه‌ای (مناسب با نوع کلاس) از نوع وکتور یا آرایه‌ی معمولی به آن طول ایجاد کرده و مقدار هر خانه را برابر با اندیس متناظرش قرار می دهد. همچنین مجموع آرایه ها را در خروجی خود قرار می دهد.

در مرحله‌ی بعدی در فایل main.cpp دو آبجکت از این دو کلاس ساخته و تابع counter() هر یک صدا می زنیم. با استفاده از کتابخانه ctime مقدار ساعت را قبل و بعد از فراخوانی تابع کانتر ذخیره نموده و با کم کردن آنها از هم و سپس تقسیم بر مقدار کلاک در هر ثانیه[۱] زمان اجرای تابع را به ثانیه بدست می آوریم. با ضرب این عدد در ۱۰۰۰۰۰۰ زمان اجرا برحسب میکروثانیه بدست می آید.

در بخش بعدی تمام مراحل بالا را در یک تابع به نام runTime() انجام می دهیم که ورودی های آن به ترتیب آبجکت موردنظر، پوینتر به تابع مورد نظر و عدد ورودی (سایز آرایه) است. توجه شود که با اینکه ساختار این تابع برای دو کلاس فوق یکی است اما نوع ورودی ها تفاوت دارد می بایست دو تابع تعریف می کردیم ولی در اینجا با استفاده از تمپلیت ها، تنها یک تابع تعریف شده است.

فقط یک نکته ی دیگر باقی می ماند و آن این است که برای فرستادن تابع counter() به تابع runTime() باید آن را به صورت پوینتر به تابع[۲] و بفرستیم که برای این منظور باید آن را به صورت رفرنسی و با مشخص کردن تایپ

[۱] CLOCKS_PER_SEC.

[۲] Pointer to function.

کلاس ارسال نماییم. که ما با تعریف تایپ مورد نظر آن را به صورت ذیل ارسال نمودیم: (توجه نمایید که باید آدرس تابع را در این متغیر بریزیم.)

```
        ...

typedef long long (LibVec::*pointer_to_vec_counter) (long long);
typedef long long (LibArr::*pointer_to_arr_counter) (long long);

        ...

int main() {

        ...

        pointer_to_vec_counter ptrVecCount = &LibVec::counter;
        pointer_to_arr_counter ptrArrCount = &LibArr::counter;

        ...
}
```

۳ سوال سه

در ابتدا خط اول فایل داده شده را می خوانیم که متوجه شویم، تعداد لیست داده شده چند عدد است. سپس با استفاده از یک حلقه هر خط را خوانده و مقادیر خوانده شده از فایل را به صورت جداگانه داخل متغیرهای مربوطه می ریزیم. (از کارکترهای [و] در ابتدا و انتهای تاریخ و ساعت صرف نظر می نماییم و آنها را داخل یک متغیر تمپ می ریزیم.)

بعد از گرفتن مقدار روز، ساعت، شماره محصول و شماره مشتری و ریختن آنها در متغیرهای مربوطه در همان شمارنده کار پردازش را نیز انجام می دهیم به این صورت که در وکتور days با استفاده از تابع `find()` در کتابخانه `algorithm` بررسی می کنیم که آیا روز دریافتی در این وکتور وجود دارد یا خیر. اگر وجود داشته باشد، پرچم[۳] تکرار را یک قرار داده و در غیر اینصورت مقدار آن را صفر یا غلط قرار می دهیم.

در قسمت بعد بر اساس مقدار پرچم تکرار (`repeatFlag`) دو کار متفاوت انجام می دهیم. در صورتی که روز جدید باشد و تکراری نباشد، باید ابتدا آن را به لیست روزها (`days`) اضافه نماییم و سپس در لیست های `productCounts` و `customerCounts` مقدار یک را قرار دهیم. یعنی یک مورد از هر کدام رخ داده است. در اینجا بعد از اضافه شدن مقدار جدید اندیس کلی هم یکی اضافه می کنیم زیرا در صورت تکراری بودن روز به مقدار این اندیس نیاز خواهیم داشت.)

[۳] Flag.

```
<!-- Page Layout here -->
<div class="row">
    <div class="col s12 m4 l3"> <!-- Note that "m4 l3" was added -->…
    </div>

    <div class="col s12 m8 l9"> <!-- Note that "m8 l9" was added -->…
    </div>
  </div>
```

شکل ۱: سوال چهار: تفکیک دو ستون کنار هم

توجه نماییم که در هر مرحله از حلقه کلی، شماره محصول و مشتری را به ترتیب در `productItems` و `customerItems` ذخیره می نماییم. (در صورتی که روز جدیدی داشته باشیم این دو لیست خالی می شوند.)

در صورتی که روز خوانده شده، تکراری باشد، دیگر نیازی نیست که وکتور روزها چیزی اضافه شود تنها باید پس بررسی مقدار متناظر (با استفاده از اندیس کلی موجود) را در `productCounts` و `customerCounts` اضافه نماییم. برای این کار مجدد با استفاده از تابع `find()` تکراری بودن مقدار جدید شماره محصول را بررسی می نماییم و در صورت غیرتکراری بودن، شمارنده متناظر را در `productCounts` یکی اضافه می نماییم. مشابه همین روند را برای شماره مشتری ها نیز انجام می دهیم. توجه نمایید که دو وکتور `productItems` و `customerItems` برای بررسی تکراری بودن مشتری و محصولات استفاده شد که اگر روزی غیرتکراری بود، مقدار این دو از اول صفر می شود.

در پایان سه لیست از نوع وکتور داریم که در آنها اطلاعات خلاصه شده قرار گرفته است و به سادگی آنها را در فایل جدید ذخیره می نماییم.

۴ سوال چهار

در این سوال ابتدا صفحه را به سه قسمت کلی تقسیم می نماییم. یک قسمت برای هدر یا همان NavBar که در آن لوگوی تلگرام و منوی همبرگری قرار می گیرد. (قسمت بالای صفحه)

قسمت پایینی نوار بالایی را به دو ستون تقسیم می نماییم. به این صورت که یک ستون را برای نام مخاطبین کوچکتر و یک قسمت را برای چت کردن با یک مخاطب خاص، بزرگتر قرار می دهیم.

در هر یک از دو ستون ایجاد شده، مجدد یک ردیف[۴] ایجاد نموده و در آنها نوار جستجو (در ستون اول و کوچکتر) و نام مخاطب مورد نظر (در ستون دوم و بزرگتر) به ترتیب قرار می دهیم.

توجه نمایید که این قسمت بندی کوچکتر و بزرگتر در صفحه های بزرگ و متوسط است و در صحفه های کوچک، این دو ستون ذیل هم قرار خواهند گرفت. (ریسپانسیو)

[۴] Row.

در قسمت کوچکتر (اولی) دو منوی پایین بازشو قرار می دهیم که با کلیک روی آنها لیست مخاطبان دیده می شود و با حرکت موس روی هر مخاطب، هاور نیز دیده می شود. (دو دکمه داریم یکی برای مخاطبان اخیر و یکی برای تمام مخاطبان)

در قسمت بزرگتر (دومی) محل چت، ابتدا یک فضا برای قرارگیری محتوای مکالمه و در پایین آن یک ورودی متن برای وارد کردن متن چت قرار می دهیم. برای قرار گیری دو شکل در کنار محل ورودی متن، نیز آنها را پشت سر هم قرار داده و مارجین ورودی متن در قسمت پایین را در فایل style.css بیشتر قرار می دهیم تا شکل ها روی متن قرار نگیرند. در ضمن موقعیت دو شکل در پایین باید به صورت نسبی باشد که کنار هم قرار گرفته و روی هم قرار نگیرند.

به این ترتیب شکلی تقریبا مشابه آنچه از ما خواسته شده را پیاده سازی نمودیم. لازم به ذکر است که منوی همبرگری، لیست های مخاطین و دو ورودی برای جستجو و متن چت، همگی از آبجکت ها و کدهای متریالایز هستند. حتی ایموجی ها نیز از گوگل فونت ها به صورت آنلاین برداشته می شوند. (برای دیدن درست صحفحه اتصال به اینترنت لازم است.) البته این امکان هم وجود دارد که فونت ها رو دانلود کرده و به صورت لوکال هم داشته باشیم ولی راه خوبی نیست زیرا اگر فونت ها توسط پشتیبانی گوگل به روزرسانی شوند و بهبود یابند دیگر ما از آنها محوریم ولی با استفاده آنلاین از فونت ها به سادگی همواره به فونت های آپدیت دسترسی داریم.

۵ ارسال روی گیت

برای ارسال روی گیت، مطابق آنچه در کلاس گفته شد ابتدا یک ریپازیتوری[۵] به نام درس برنامه سازی پیشرفته درست کردیم و سپس در سیستم لوکال خود آن را دانلود کرده و فایل های مورد نظر را با پوشه بندی مناسب در دایرکتوری ایجاد شده قرار دادیم. با استفاده از دستور

```
git add DIRECTORYofFILES
```

فایل ها را اضافه کرده و سپس با دستور

```
git commit -m "MYCOMMENT"
```

آنها را روی سیستم لوکال کامیت یا ثبت می کردیم. حال با دستور

```
git push origin master
```

فایل های کامیت شده را روی سرور github.com و روی برنچ مستر بارگذاری نمودیم.

توجه نمایید که در ابتدا برای گرفتن ریپو از دستورهای ذیل به ترتیب استفاده نمودیم. (از طریق ssh)

```
git init
git clone git@github.com:SRHashemirad/AUT_AP_course.git
```

[۵] Repositoriy.

همچنین قابل ذکر است که فایل `.gitignore` را به به صورتی قرار می دهیم که فایل هایی که در حین کامپایل ایجاد می شوند[۶] را ثبت نکرده و همچنین نسخه هایی که نرم افزار emacs ایجاد می کند[۷] را نیز ثبت ننماید.

شما می توانید برای دستیابی به فایل ها در گیت هاب از این لینک اقدام نمایید.

با تشکر فراوان از حسن دقت نظر و توجه شما

[۶] *.o

[۷] *.*~