نویسنده احمق

یک نویسنده احمق در یکی از انتشارات بزرگ یک کشور انگلیسی زبان استخدام شده است. او فقط بلد است داستان از خودش در بیاورد (ببافد) ولی حتی توانایی تشخیص این که ژانر این داستان چیست را هم ندارد. برای مدتی داستان ها را به سلمان و دانیال می‌داد که آن‌ها بخوانند و موضوع داستان را مشخص کنند. بعد چند ماه که سلمان و دانیال دیگر به امتحانات پایان ترم نزدیک می‌شدند و حجم کارهای درسی و حتی غیر درسی‌شان (!) زیاد شده بود، تصمیم گرفتند که به جای اینکه وقتشان را صرف خواندن داستان‌های غالبا پرت و پلای نویسنده احمق کنند، این کار بیهوده را به دانشجوهای درس برنامه نویسی بسپارند! اما از طرفی چون درصدی از سود فروش کتاب‌ها می‌گرفتند و می‌خواستند که برای همیشه از این سود فروش کتاب‌های بعدی نیز برخوردار شوند، پروژه ای را تعریف کردند که به طور اتوماتیک ژانر داستان ها را تشخیص دهد تا ترم‌های بعد نیز از برنامه آن استفاده کنند. بنابراین، شما به عنوان دانشجوی درس برنامه نویسی باید یک برنامه c++ با ویژگی های خواسته شده در زیر بنویسید.

در این پروژه هدف این است که ژانر تعدادی داستان را تشخیص دهیم. به این صورت که داستان ها در قالب فایل های .txt هستند و تعدادی کلمات مرتبط با هر ژانر نیز در فایل های .csv به طور مجزا تعریف شده اند. برای تشخیص ژانر هر داستان، لازم است که تعداد کلمه های هر داستان که متعلق به هر کدام از ژانر ها شمرده شوند. مشخصا داستان مربوط به ژانری است که بیشترین تعداد کلمات از آن ژانر را داشته باشد. در ادامه جزئیات برنامه توضیح داده می شود.

بارگیری کلمات هر ژانر

در ابتدای شروع برنامه، لازم است اطلاعات ژانر ها از فایل های .csv که نام هر فایل از قبل مشخص شده است، خوانده شوند و لیست کلمات کلیدی هر ژانر در حافظه برنامه ذخیره شوند تا برای استفاده های بعدی مورد استفاده قرار گیرد. به طور کلی چهار ژانر عاشقانه، معمایی، فانتزی و علمی-تخیلی را میخواهیم بررسی کنیم که اسم فایل کلمات کلیدی هر کدام از این ژانرها به صورت زیر خواهد بود.

- Romance.csv
- Mystery.csv
- Fantasy.csv
- SciFi.csv

در هر فایل کلمات کلیدی اطلاعات هر کلمه کلیدی شامل نام کلمه کلیدی و ضریبی به عنوان وزن آن کلمه برای ژانر مورد نظر در خط های فایل ورودی آمده است. فرمت فایل کلمات کلیدی یک ژانر به صورت زیر خواهد بود:

```
Keyword, Weight
{keyword}, {weight}
…
```

توجه داشته باشید که در شروع برنامه لازم است که فایل‌های کلمات کلیدی بارگیری شوند چرا که در ادامه برنامه نیاز است که استفاده شوند. در صورتی یک یا چند تا از فایل‌ها وجود نداشت یا مشکلی در خواندن فایل‌ها پیش آمد پیغام زیر نمایش داده شود و برنامه اجرا نشود.

Error importing genre keywords. Please check keyword files.

شروع برنامه

با اجرا کردن برنامه، ابتدا کلمات هر ژانر خوانده شده و دستورات توسط کاربر وارد می‌شوند. در زیر هر دستور توضیحی در مورد آن دستور نیز ارائه شده است.

- **show_the_list_of_commands**

لیست تمام دستورات را پرینت می‌کند.

- **import_story {story_name.txt}**

نام فایل داستان را می‌گیرد و در صورت وجود داشتن آن فایل، آن را می‌خواند و به داستان های خوانده شده اضافه می‌کند. توضیحات دقیق این دستور در صفحات بعد توضیح داده می‌شود.

- **show_the_list_of_stories**

لیست داستان‌های خوانده شده را به ترتیب زمان بارگیری و با شماره‌شان را نمایش می‌دهد. نحوه نمایش لیست داستان ها مطابق زیر خواهد بود.

```
List of all imported stories:
1. Story1
2. Story2
3. Story3
...
```

دقت کنید که حروف اول نام داستان‌ها بزرگ است و مابقی حروف کوچک اند.

- **analyze_story {story_index} {output_file_name.txt}**

شماره داستان مورد نظر را دریافت می‌کند و پس از آنالیز داستان، نتایج را در فایل txt مشخص شده دریافت می‌کند. نحوه تحلیل داستان و فرمت فایل خروجی در ادامه توضیح داده می‌شود. در صورتی که {story_index} داده شده صحیح نباشد پیغام زیر نمایش داده می‌شود.

Invalid story index.

در دستورهای نیز در صورتی که نیاز به ورودی دادن {story_index} به دستوری بود، این حالت خطا را نیز لحاظ کنید.

- **analyzed_stories_list**

لیست داستان‌هایی که آنالیز شده‌اند را در ترمینال به صورت زیر نمایش می‌دهد.

```
The analyzed stories are: Story1_name, story2_name, ... and story9_name.
```

نام اولین داستان با حرف بزرگ شروع می‌شود و قبل از آمدن نام آخرین داستان از and استفاده می‌شود.

- **show_story_analysis {story_index}**

این دستور با دریافت شماره داستان {story_index}، دقیقا همان نتایجی که دستور analyze_story در فایل خروجی‌اش نوشته است را در ترمینال به کاربر نمایش می‌دهد (برنامه نتایج را از حافظه تولید می‌کند و از فایل ذخیره شده استفاده نمی‌کند). در صورتی که داستان آنالیز نشده بود پیغام زیر نمایش داده شود:

This story has not been analyzed yet. Please use the analyze_story command.

- **dump_analyzed_stories {output_file_name.csv}**

همه داستان‌های خوانده شده را مطابق با فرمتی که در ادامه توضیح داده می‌شود در یک فایل .csv که نام فایل در ادامه دستور آمده است ذخیره می‌کند و سپس پیغام زیر را نمایش می‌دهد.

All analyzed stories dumped in {output_file_name.csv}.

در صورتی که فایلی با نام وارد شده از قبل وجود داشت، فایل قبلی را پاک کرده و از ابتدا خروجی‌های جدید را در آن فایل ذخیره می‌کند.

در صورتی که هنوز داستانی آنالیز نشده بود پیغام زیر نمایش داده شود.

No analyzed stories to dump.

در فایل خروجی اطلاعات زیر چاپ می‌شود:

Story, Genre, Confidence, Romance Words, Mystery Words, Fantasy Words, SciFi Words, Common Keyword 1, Common Keyword 2, Common Keyword 3, Common Keyword 4
{story_name}, {confidence}, {romance_words}, {mystery_words}, {fantasy_words}, {scifi_words}, {common_keyword1}, {common_keyword2}, {common_keyword3}, {common_keyword4}
…

دقت کنید که خط اول ثابت است و عنوان ستون‌های جدول را شامل شده است. در خط های بعدی به تعداد داستان‌های خوانده شده اطلاعات هر داستان نمایش داده می‌شود. در ستون اول نام داستان، در ستون دوم ژانر تشخیص داده شده برای آن داستان و در ستون سوم درصد اطمینان آن ژانر نوشته می‌شود. در ستون‌های چهارم تا هفتم تعداد کلماتی از داستان که متعلق به هر ژانر است نمایش داده می‌شود و در ستون‌های هشتم تا یازدهم نیز پرتکرار ترین کلمه‌های کلیدی ژانر اصلی داستان به ترتیب از بیشترین تا کمترین نوشته می‌شود.

- **exit**

در صورت وارد شدن این دستور، برنامه خاتمه می‌یابد.

بارگیری داستان

این عملیات که توسط دستور import_story فراخوانی می‌شود، با گرفتن نام فایلی که داستان مورد نظر در آن قرار دارد آن را بارگیری می‌کند. برای سادگی فرض کنید که فایل مورد نظر در فولدر فعلی (همان فولدری که فایل .exe برنامه در آن در حال اجرا شدن است) قرار دارد. با اجرای این عملیات برنامه به دنبال فایل مورد نظر می‌گردد. در صورتی که فایل وجود داشت و بدون مشکل محتوای داستان خوانده شد، پیغام زیر را نمایش می‌دهد:

{story_name} imported successfully.

سپس نام داستان به لیست داستان‌های خوانده شده اضافه می‌شود.

در صورتی که فایل مورد نظر وجود نداشت پیغام زیر نمایش داده شود:

File not found.

در صورتی که فایل وجود داشت اما امکان خواندن فایل (به هر دلیلی) با مشکل مواجه شد پیغام زیر نمایش داده شود:

Error importing the story.

تشخیص ژانر داستان

با اجرای این عملیات که توسط دستور analyze_story فراخوانی می‌شود، برنامه به ترتیب همه کلمات داستان را با مجموعه کلمات کلیدی هر ژانر مقایسه کرده و تعداد کلماتی که در داستان متعلق به هر ژانر است را محاسبه می‌کند. در نهایت ژانری که بیشترین درصد کلمات را به خود اختصاص داد، ژانر تشخیص داده شده برای آن داستان خواهد بود.

پس از تشخیص ژانر داستان مورد نظر، خروجی به صورت زیر در ترمینال نمایش داده می‌شود.

The genre of the story {story_name} is {genre}.

که عبارت {story_name} نام داستان خوانده شده است و عبارت {genre} ژانر تشخیص داده شده برای داستان است. همچنین اطلاعات آماری زیر در فایل با نام داده شده در دستور {output_file_name.txt} ذخیره می‌شود. در صورتی که فایلی با نام وارد شده از قبل وجود داشت، فایل قبلی را پاک کرده و از ابتدا خروجی‌های جدید را در آن فایل ذخیره می‌کند.

```
Story Name: {story_name}
Predicted Genre: {genre}


Genre, Number of Keywords, Confidence
Romance, {n_romance}, {confidence_romance}
Mystery, {n_mystery}, {confidence_mystery}
Fantasy, {n_fantasy}, {confidence_fantasy}
SciFi, {n_scifi}, {confidence_scifi}
```

محاسبه درصد هر ژانر

برای محاسبه درصد هر ژانر، فرض کنیم مجموع وزن‌های کلمات داستان که مربوط به ژانر i م اند، برابر با m_i باشد. بنابراین داریم:

$$m_i = \sum_{j=1}^{N} w_j n_j$$

که در این رابطه w_j وزن کلمه j م و n_j تعداد تکرار کلمه j م در داستان و N تعداد کلمات موجود در داستان از ژانر i م است.

حال شانس اینکه داستان مربوط به ژانر i م باشد برابر خواهد بود با:

$$p_i = \frac{m_i}{\sum_{i=1}^{4} m_i}$$

بدیهی است که مجموع درصدهای همه ژانر ها برابر با ۱۰۰ درصد خواهد شد.

پرتکرارترین کلمه‌های کلیدی

برای هر داستان نیاز است که پنج کلمه کلیدی پر تکرار را پیدا کنیم. کلمات کلیدی پرتکرار حتما باید از میان کلماتی باشد که مربوط به ژانر اصلی داستان هستند و بیشترین تکرار را در طول متن داشته باشند. مثلا کلمه مریخ ممکن است در یک داستان علمی تخیلی پر تکرار ترین کلمه کلیدی باشد. این پنج کلمه کلیدی پر تکرار نیز باید در انتهای فایل خروجی داستان به صورت زیر ذخیره شود.

```
The common keywords of the story are: {keyword1}, {keyword2}, {keyword3}, {keyword4}, {keyword5}.
```

(امتیازی) پیدا کردن اسم شخصیت اول

این عملیات که توسط دستور زیر فراخوانی می‌شود، علاوه بر دستورات بالا در لیست دستورات برنامه قرار می‌گیرد.

- **find_the_first_character {story_index}**

در صورتی که {story_index} داده شده صحیح نباشد پیغام زیر نمایش داده می‌شود.

```
Invalid story index.
```

از آن‌جایی که اسامی خاص با حرف بزرگ شروع می‌شوند، از میان کلماتی که با حرف بزرگ شروع شده اند کلمه‌ای به عنوان اسم شخصیت اول انتخاب می‌شود که بیشترین تکرار را در متن داشته باشد. احتمال اینکه کلمه‌ای وجود داشته باشد که بیشتر از یک بار در ابتدای جمله‌ای وجود داشته باشد (که حرف اول آن بزرگ باشد) می‌توان با دقت خوبی انتظار داشت که برنامه می‌تواند نام شخصیت اول را به درستی تشخیص دهد. پس از پیدا کردن نام شخصیت اول پیغام زیر در ترمینال نمایش داده می‌شود.

```
The first character of {story_name} is {character_name}.
```

در صورتی که هنوز داستان مورد نظر آنالیز نشده بود پیغام زیر نمایش داده شود.

This story has not been analyzed yet. Please use the analyze_story command.

نکاتی در مورد وارد کردن دستورها

در رابطه با دستورهایی که نیاز به دریافت ورودی دارند در صورتی که کاربر فقط نام دستور را وارد کرد راهنمای استفاده صحیح از آن دستور مطابق با آنچه پیشتر اشاره شود نمایش داده می‌شود. به عنوان مثال اگر کاربر عبارت زیر را به عنوان ورودی وارد کرد:

analyze_story

عبارت زیر نمایش داده شود:

“analyze_story {story_index} {output_file_name.txt}”

("ها (دبل‌کوتیشن‌ها) به این معنا اند که دقیقا این عبارت باید نمایش داده شود و در خروجی نهایی " نمایش داده نمی‌شود) یا عنوان مثالی دیگر اگر کاربر تنها عبارت زیر را ورودی داد:

dump_analyzed_stories

عبارت زیر به عنوان راهنما نمایش داده می‌شود:

“dump_analyzed_stories {output_file_name.csv}”

در صورتی که دستوری به جز دستورهایی که در لیست کامندها ذکر شده است توسط کاربر وارد شد عبارت زیر نمایش داده شود.

Command not found. See the list of commands with show_the_list_of_commands.

نکات پایانی

- کد شما با تستر نمره داده می‌شود و در صورتی که ورودی‌ها و خروجی‌های کد شما مطابق با موارد گفته شده نباشد نمره‌ای به فایل تحویل داده شما تعلق نمی‌گیرد. دقت داشته باشید که کد شما تنها در یک فایل از طریق سایت کوئرا دریافت می‌شود و الزاما کد شما باید در کوئرا به درستی کامپایل شود. بنابراین استفاده از کتاب‌خانه‌هایی نظیر conio.h و... مجاز نیست. استفاده از vector, string, struct و سایر مباحثی که در کلاس درس تدریس شده است مجاز است.
- تمیزی کد، ذخیره کردن اطلاعات در ساختارهای مناسب، شکستن مرحله‌به‌مرحلهٔ مسئله و طراحی مناسب برنامه، در کنار تولید خروجی دقیق و درست، بخش مهمی از نمره شما را تعیین خواهد کرد.
- در کنار این فایل، یک یا چند ویدئو نیز برای واضح شدن مطالب و توضیح در مورد نحوه کارکرد برنامه در اختیار شما قرار داده می‌شود. مجموع این سند و ویدئوی توضیحات و اطلاع‌رسانی‌های کانال درس، مراجع اصلی پروژه هستند.