سیدمحمدامیر خدامی ۹۵۳۶۱۸۰۴۳۰۰۴ سیدعلی میرامیرخانی ۹۵۳۶۱۸۰۴۳۰۱۰

پروژه اول درس مبانی بازیابی اطلاعات و جست‌وجوی وب استاد درس خانم دکتر باباعلی

۱- پوشه‌ها

۱-۱- ریشه

- دارای فایل‌های اصلی اجرای برنامه شامل:
- `main.py`: فایل اصلی اجرای برنامه که برای پردازش متن‌ها، تولید جدول لغات، دریافت و پردازش کوئری و نمایش پاسخ به کار می‌رود.
- `getDocText.py`: یک ماژول جانبی برای برنامه اصلی (`main`) که وظیفه خواندن فایل‌ها و برگرداندن لیست نام فایل‌ها به همراه محتوای داخل آن را بر عهده دارد.
- `positionalIndex.py`: یک `dataObject` است که به ازای هر `term` یک دیکشنری ایجاد می‌کند. (این دیکشنری تمام رخدادهای `term` در هر سند و موقعیت‌های رخداده در آن سند را ذخیره می‌کند.)
- `CreateText.py`: برنامه‌ای مجزا که به شما امکان افزودن سند به سندهای موجود را می‌دهد. این برنامه متن عنوان درخواستی را از ویکی‌پدیا گرفته وبا فرمت متنی در پوشه `./dosc` ذخیره می‌کند.

۱-۲- `docs`

این پوشه تمام سندهای مورد پردازش را در خود جای داده است. با هر بار اجرای برنامه اصلی تمام سندهای این پوشه `index` شده و `indexTable` نظیر، از روی آن ساخته می‌شود. می‌توانید مستندهای موردنظر خود را برای تست داخل این پوشه قرار دهید.

۱-۳- `logs`

با هر بار اجرای برنامه تاریخچه کاملی از مستندات موجود در لحظه اجرا، `indexTable` ایجاد شده، کوئری و نتیجه آن تحت یک فایل متنی به نام تاریخ اجرای برنامه در این پوشه ذخیره خواهد شد. می‌توانید با بررسی این لاگ‌ها صحت برنامه را بررسی کنید.

۲- فرمت قابل قبول

- تنها operand های NEAR یا WITH یا AND-OR-NOT در کوئیری قابل قبول است. ترکیب هر کدام از این سه مجموعه پاسخی به همراه نخواهد داشت.
- منطق اولویت برنامه از چپ به راست و برای دسته AND-OR-NOT اعمال می‌شود.
- این موتور جست‌وجو برای اجرا نیاز به دقیقا سه ورودی به عنوان term دارد.
- Operandهای بولی باید دقیقا به صورت uppercase نوشته شده و term ها به صورت lowercase وارد شوند.
- در استفاده از عملگر NEAR باید میان عدد و کلمه NEAR یک فاصله وجود داشته باشد، به صورت a NEAR 3 b
- نمونه‌هایی از کوئری‌های قابل قبول:

```
1. a OR b AND c
2. a WITH b WITH c
3. a NEAR 3 b NEAR 4 c
4. A OR NOT b AND C
```

- نکته: در هنگام نمایش termهای شناخته شده موجود در query عملگر not نیز در صورت وجود در لیست، در کنار یک یا دو term قابل مشاهده است، اما این بدین معنا نیست که این کلمه به عنوان عملگر شناخته نشده بلکه چون تک عملوندی است در آخر بروی آن عملوند اجرا می‌شود.

۳- منطق برنامه

- کار با اجرای ماژول جانبی getDocText آغاز شده و محتوای سندها و نام‌آنها باز می‌گردد، سپس به هر سند یک آیدی اختصاص داده می‌شود و محتوای داخل هر سند پاکسازی می‌شود.
- پاکسازی شامل حذف سیمبل‌های زائد نظیر « !()-[]{};:\'",<>./?@#$%^&*_~ » و یکدست سازی حروف به صورت lowercase است.
- سپس از روی محتوای این سندها یک indexTable تولید می‌شود که شامل termها به همراه سندهای شامل آن و مکان‌های حضور term در هر سند است.
- کوئری نیز از کاربر گرفته شده، termهای آن از هم جدا شده، عملگرهای NEAR, AND, OR, WITH از داخل آن استخراج می‌شوند و دیگر termهای موجود برای یکنواختی به حالت lowercase تبدیل می‌شوند.

- اگر ترکیب ورودی به ازای کوئری صحیح باشد یکی از دو تابع `calc_and_or()` برای ترکیب‌های AND-OR-NOT و `calc_with_near()` برای ترکیب‌های فقط NEAR یا فقط WITH اجرا خواهند شد.
- در تابع `calc_and_or()` ابتدا عملگر NOT در صورت وجود، بروی عملوند مدنظر به کمک تابع `notdef()` اجرا سپس با کمک `anddef()` و `ordef()` کوئری مورد پردازش قرار گرفته و خروجی حاصل می‌شود.
- در تابع `calc_with_near()` نیز در صورت وجود داشتن عملگر WITH تابع `withdef()` فراخوانده شده و با جست‌وجوی `indexTable` موجود نتیجه برمی‌گردد. اگر عملگر NEAR باشد نیز تابع `neardef()` فراخوانده می‌شود و با دو عدد گرفته به عنوان متغیر NEAR جدول `indexTable` را جست‌وجو کرده و نتیجه را برمی‌گرداند.
- در طول اجرای برنامه نیز تمام اطلاعات از استخراج سندها تا نتایج به دست آمده در لاگ فایل نظیر اجرا ذخیره می‌شود.