به نام خدا

دانشکده علوم و فنون نوین

آشنایی با زبان‌شناسی رایانشی (۱۲۱-۰۵-۸۳)
نیم‌سال دوم ۱۴۰۱-۱۴۰۰

تاریخ تحویل: ۱۴۰۰/۱۲/۲۷

تمرین شماره ۱

---

۱. (۱۰٪) **[پژوهش: دادگان‌های زبانی]** هدف این تمرین آشنایی با دادگان‌های زبانی و ساختار آنهاست. هر دانشجو حداقل دو مورد از موارد زیر را انتخاب کرده و آنها را بررسی کنند. نتیجه بررسی خود را شامل توضیحات دادگان، کاربرد(های) آن، اطلاعات آماری مرتبط (مانند حجم و ...)، مرجع جمع‌آوری و منابع مورد استفاده خود را گزارش کنید.

برای یافتن اطلاعات مورد نیاز می‌توانید به منابع دادگانی شناخته شده زیر و یا گوگل! مراجعه کنید:

https://catalog.ldc.upenn.edu
http://catalog.elra.info/en-us
http://www.peykaregan.ir

توجه کنید برخی از این دادگان‌ها دارای چند نسخه هستند و در صورت انتخاب هر کدام لازم است همه نسخه‌ها بررسی شده و تفاوت آنها نیز آورده شود:

۱. TIMIT
۲. فارس دات
۳. Resource Management
۴. ATIS
۵. WSJ
۶. CALLHOME
۷. CALLFRIEND
۸. PhoneBook
۹. Switchboard
۱۰. HUB5
۱۱. CSLU
۱۲. NIST Speaker Recognition Evaluation
۱۳. GALE
۱۴. GlobalPhone
۱۵. Treebank
۱۶. UN Parallel Text
۱۷. فارس نت
۱۸. MIZAN Persian-English Parallel Corpus
۱۹. پیکره همشهری
۲۰. BBN Pronoun Coreference and Entity Type Corpus
۲۱. مجموعه داده عروض
۲۲. لغت‌نامه احساس لِکسی‌پِرس(LexiPers)

h.veisi@ut.ac.ir

به نام خدا

۲۳. واژگان زایای زبان فارسی

۲۴. پرسیکا

۲۵. پیکره درختی وابستگی فارسی اوپسالا

۲۶. سِنتی‌پِرس (SentiPers)

۲۷. پیکره وابستگی نحوی زبان فارسی

---

۲. (۴۰٪) **[پیاده‌سازی]** هدف از این تمرین آشنایی با شیوه کار با فایل‌های متنی و پیش‌پردازش اولیه متون فارسی می‌باشد. به همراه این تمرین ۷۰ فایل با فرمت txt ارائه شده که هر کدام حاوی یکی از متون خبری خبرگزاری ایسنا می‌باشد. فایل‌ها در ۷ عنوان خبری (هر کدام ۱۰ فایل) که هرکدام در یک پوشه قرار دارند، دسته‌بندی شده‌اند. از این به بعد، به کل این متن (۷۰ فایل) پیکره متنی تمرین‌های درس زبان‌شناسی رایانشی (**زبرا**) می‌گوییم.

برای موارد زیر یک برنامه (به زبان برنامه‌نویسی دلخواه) بنویسید. در هر بخش از تمرین، خروجی برنامه را به همراه کد برنامه نوشته شده و گزارش آن، به تفکیک هر بخش از سوال، ارسال کنید. دقت کنید که تمامی توابع کد خود را به تفکیک و با نام مناسب نام‌گذاری کنید که در تمرین‌های بعدی نیز قابل استفاده باشد.

الف) تابعی بنویسید که یک فایل متنی را گرفته و موارد زیر را به عنوان خروجی برای آن محاسبه کند:

۱) تعداد کل کلمات

۲) تعداد کل کلمات یکتا

۳) تکرار هر کدام از کلمات یکتا.

نتیجه این بخش را برای برای اولین فایل هر عنوان خبری در گزارش خود بیاورید.

ب) با استفاده از تابع نوشته شده در مرحله الف، یک تابع دیگر بنویسید که یک پوشه (حاوی ۷ پوشه دیگر) را به عنوان ورودی بگیرد و همه فایل‌های همه پوشه‌ها را بررسی کرده و در پایان آمار زیر را

h.veisi@ut.ac.ir

به نام خدا

استخراج کند:

۱) تعداد کل کلمات هر پوشه (موضوع)

۲) تعداد کل کلمات یکتای هر پوشه

۳) تعداد کل کلمات پیکره زبرا

۴) کل کلمات یکتای پیکره زبرا.

آمارهای فوق را در گزارش خود ارائه کنید.

ج) یک تابع جدید بنویسید که یک پوشه را به عنوان ورودی گرفته و تمام فایل‌های موجود در آن را با یکدیگر ادغام کند به گونه‌ای که هر سطر فقط حاوی مطالب یکی از فایل‌ها باشد. این تابع را برای همه ۷ پوشه مختلف پیکره زبرا استفاده کنید و خروجی برنامه خود را ارسال کنید.

حال همان تابع را روی ۷ فایل بدست آمده از مرحله قبل اعمال کنید تا کل پیکره زبرا در یک فایل متنی قرار گیرد. فایل نهایی را به همراه پاسخ تمرین ارسال کنید.

بعد از انجام این تمرین شما باید توابع زیر را اختیار داشته باشید.

توابع سوال ۲ تمرین شماره ۱
NoAllWords , NoAllUniqWords, NoEachUniqWord = **CountWordsFile**(TextFileName) تابعی که فایل متنی (TextFileName) را گرفته و تعداد کل کلمات (NoAllWords)، تعداد کل کلمات یکتا (NoAllUniqWords) و تکرار هر کدام از کلمات یکتا (NoEachUniqWord) را به عنوان خروجی برگرداند.
NoAllWordsPerFolder , NoAllUniqWordsPerFolder, NoEachUniqWordPerFolder, NoEachUniqWord = **CountWordsFolder**(Path) تابعی که مسیر (Path) حاوی یک یا چند پوشه را گرفته و تعداد کل کلمات هر پوشه (NoAllWordsPerFolder)، تعداد کل کلمات یکتا در هر پوشه (NoAllUniqWordsPerFolder)، تکرار هر کدام از کلمات یکتا در هر پوشه (NoEachUniqWordPerFolder) و تکرار هر کدام از کلمات یکتا در کل پوشه‌های موجود در مسیر داده شده (NoEachUniqWord) را به عنوان خروجی برگرداند.
OutFile = **CombineFiles** (Path) تابعی که مسیر (Path) را گرفته و تمام فایل‌های موجود در آن را با یکدیگر ادغام کند به گونه‌ای که هر سطر فقط حاوی مطالب یکی از فایل‌ها باشد. خروجی این تابع در متغیر OutFile قرار داده می‌شود.

h.veisi@ut.ac.ir

به نام خدا

۳. (۵۰٪) **[پیاده‌سازی: نرمال‌سازی و واحدسازی]** هدف این تمرین پردازش متون فارسی به منظور نرمال‌سازی و واحدسازی آن می‌باشد. در این سوال نیز از دادگان **زبرا** که در سوال ۲ ارائه شد، استفاده کنید و برای بخش‌های زیر یک برنامه بنویسید و در هر بخش از تمرین، خروجی برنامه را به همراه کد برنامه نوشته شده و گزارش آن، ارسال کنید.

در این تمرین فرض شده است از تمرین قبل، توابع زیر را اختیار دارید.

توابع سوال ۲ تمرین شماره ۱
NoAllWords , NoAllUniqWords, NoEachUniqWord = **CountWordsFile**(TextFileName) تابعی که فایل متنی (TextFileName) را گرفته و تعداد کل کلمات (NoAllWords)، تعداد کل کلمات یکتا (NoAllUniqWords) و تکرار هر کدام از کلمات یکتا (NoEachUniqWord) را به عنوان خروجی برگرداند.
NoAllWordsPerFolder , NoAllUniqWordsPerFolder, NoEachUniqWordPerFolder, NoEachUniqWord = **CountWordsFolder**(Path) تابعی که مسیر (Path) حاوی یک یا چند پوشه را گرفته و تعداد کل کلمات هر پوشه (NoAllWordsPerFolder)، تعداد کل کلمات یکتا در هر پوشه (NoAllUniqWordsPerFolder)، تکرار هر کدام از کلمات یکتا در هر پوشه (NoEachUniqWordPerFolder) و تکرار هر کدام از کلمات یکتا در کل پوشه‌های موجود در مسیر داده شده (NoEachUniqWord) را به عنوان خروجی برگرداند.
OutFile = **CombineFiles** (Path) تابعی که مسیر (Path) را گرفته و تمام فایل‌های موجود در آن را با یکدیگر ادغام کند به گونه‌ای که هر سطر فقط حاوی مطالب یکی از فایل‌ها باشد. خروجی این تابع در متغیر OutFile قرار داده می‌شود.

الف) (۵٪) ابتدا با استفاده از تابع CombineFiles کل فایل‌های متنی مجموعه دادگان را به یک فایل (به اسم ZebraAllRaw) تبدیل کنید و در آن با استفاده از تابع CountWordsFile تعداد کل کلمات، تعداد کل کلمات یکتا و تکرار هر کدام از کلمات یکتا را محاسبه کنید. در این بخش تعداد کل کلمات و تعداد کل کلمات یکتا را گزارش کنید.

ب) (۵٪) تابعی با نام CountChar بنویسید که یک فایل متنی و یک کاراکتر را گرفته و تعداد آن کاراکتر را در آن فایل برگرداند. با این تابع تعداد فاصله‌ها (Space) را در فایل ZebraAllRaw شمارش کنید و

h.veisi@ut.ac.ir

گزارش کنید.

ج) (۱۵٪) تابعی با نام CorrectPuncs بنویسید که یک فایل را گرفته و اشکال‌های موجود در فاصله‌گذاری علائم نگارشی علائمی مانند «. ، ؛ : ! ؟ {} «» [] ()» را اصلاح کرده و فایل جدیدی را حاوی متن اصلاح شده برگرداند. فایل ZebraAllRaw را به این تابع بدهید و خروجی آن را ZebraAllPuncs بنامید. فایل خروجی را به همراه این بخش گزارش کنید و با استفاده از تابع CountChar تعداد فاصله‌های فایل جدید را بشمارید و گزارش کنید. در مورد مقایسه تعداد کارکترهای فاصله در این بخش و بخش قبل بحث کنید.

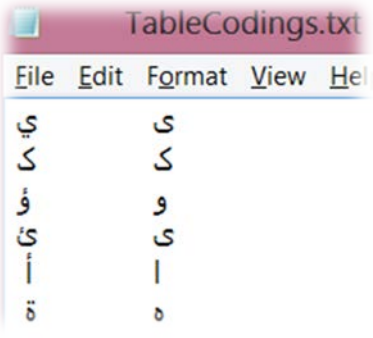
برای این بخش از عبارات منظم (Regular Expression) استفاده کنید. به عنوان مثال اگر قبل از علامت ، یک یا چند فاصله وجود داشت همه آنها را حذف کند و یا اگر پرانتز بسته به کلمه بعد از خودش چسپیده بود (به جز نقطه)، آن را از کلمه بعدی جدا کند.

در پایتون ابتدا بسته RE را import کنید و سپس از توابع آن مانند re.search(pattern, string) برای یافتن الگوی pattern (که به صورت عبارت منظم نوشته شده است) در string استفاده کنید.

د) (۱۰٪) یک تابع با نام CorrectCoding بنویسید که دو فایل متنی، یکی متنی که باید نرمال شود (مانند پیکره زبرا) و دیگری متنی که حاوی اصلاحات است (فایلی با نام TableCodings ایجاد کنید) را گرفته و اشکال‌های کدگذاری کاراکترهایی مانند «ی ؤ ئ ک» را مطابق آنچه گفته شد اصلاح کرده و فایل جدیدی را حاوی متن اصلاح شده برگرداند. فایل TableCodings حاوی دو ستون است که با Tab (\t) از هم جدا شده‌اند (شکل زیر)، یک ستون شکل درست کاراکتر و ستون دیگر شکل غلط آن است.

h.veisi@ut.ac.ir

به نام خدا



فایل ZebraAllPuncs را به این تابع بدهید و خروجی آن را ZebraAllCoding بنامید. فایل خروجی و فایل TableCodings را به عنوان پاسخ این بخش ارسال کنید.

هـ) (۱۵٪) یک تابع با نام CorrectVariations بنویسید که دو فایل متنی، یکی متنی که باید نرمال شود (مانند پیکره زبرا) و دیگری متنی که حاوی اصلاحات است (فایلی با نام TableVariations ایجاد کنید) را گرفته و اصلاحات بیان شده در TableVariations را در فایل ورودی اعمال کند و فایل جدیدی را حاوی متن اصلاح شده برگرداند. فایل TableVariations مشابه شکل زیر حاوی دو ستون است که با Tab (\t) از هم جدا شده‌اند، یک ستون شکل درست کلمه و ستون دیگر شکل غلط آن است. این فایل را بر اساس دانش خود و کلمات موجود در پیکره تکمیل کنید. بدیهی است که اشکال نیم فاصله در پسوندها (مانند «می» در «می توان») و پسوندها (مانند «ها» در «سازمانها») را نیز می‌توان با این روش حل کرد.

h.veisi@ut.ac.ir

به نام خدا

دانشکده علوم و فنون نوین

آشنایی با زبان‌شناسی رایانشی (۱۲۱-۰۵-۸۳)
نیم‌سال دوم ۱۴۰۱-۱۴۰۰

تاریخ تحویل: ۱۴۰۰/۱۲/۲۷

تمرین شماره ۱

فایل ZebraAllCoding را برای اصلاح به این تابع بدهید و خروجی آن را ZebraAllNormalized بنامید. فایل خروجی و فایل TableVariations را به عنوان پاسخ این بخش ارسال کنید. در فایل خروجی، تعداد کل کلمات، تعداد کلمات یکتا و تعداد کاراکترهای فاصله را گزارش کنید. تفاوت تعداد حاصل شده در این بخش با تعداد بخش‌های قبلی را تشریح کنید.

بعد از این تمرین، توابع زیر به مجموعه برنامه شما افزوده می‌شود.

توابع سوال ۳ تمرین شماره ۱
NoChar = **CountChar** (InTextFileName, CharName) تابعی که فایل متنی (InTextFileName) و یک کاراکتر (CharName) را گرفته و تعداد کل آن کارکترها (NoChar) را به عنوان خروجی برگرداند.
OutTextFileName = **CorrectPuncs** (InTextFileName) تابعی که فایل متنی (InTextFileName) را گرفته و پس از اصلاح اشکال‌های علائم سجاوندی در متن، فایل خروجی (OutTextFileName) را به عنوان خروجی برگرداند.
OutTextFileName = **CorrectCoding** (InTextFileName, TableCodings) تابعی که فایل متنی (InTextFileName) و فایل حاوی اصلاحات (TableCodings) را گرفته و پس از اصلاح اشکال‌های کدگذاری در متن، فایل خروجی (OutTextFileName) را به عنوان خروجی برگرداند.
OutTextFileName = **CorrectVariations** (InTextFileName, TableVariations) تابعی که فایل متنی (InTextFileName) و فایل حاوی اصلاحات (TableVariations) را گرفته و پس از اصلاح اشکال‌های تنوع املایی در متن، فایل خروجی (OutTextFileName) را به عنوان خروجی برگرداند.

h.veisi@ut.ac.ir