\documentclass[a4paper,10pt]{article}

\usepackage{etex} % fix some out-of-resource errors of latex

\usepackage{graphicx}

\usepackage{amsmath}

\usepackage{amssymb}

\usepackage{paralist}

\usepackage{xspace}

\usepackage[utf8]{inputenc} % for turkish characters in UTF8

\usepackage{booktabs}

\usepackage{multirow}

\usepackage{url} % \url

\usepackage{hyperref}

% adjust these values if your printer cannot print so far

%\usepackage[left=10mm,right=10mm,top=20mm,bottom=10mm,headheight=13pt,headsep=8pt]{geometry}

\usepackage[left=20mm,right=20mm,top=20mm,bottom=15mm,headheight=13pt,headsep=8pt]{geometry}

\usepackage{fancyhdr}

\pagestyle{fancy}

\fancyhf{}

\fancyhead[L]{Object Oriented Software Design --- Marmara Üniversitesi}

\fancyhead[R]{Python Programming Assignment}

%\input{macros}

\newcommand\bi{\begin{compactitem}}

\newcommand\ei{\end{compactitem}}

\begin{document}

\noindent

The program `analyze.py' (see homepage) reads a file,

splits it into words,

counts which sequences of length 2 and 3 of letters appear in words how often,

and then displays the 5 most frequent sequences.

%

In this assignment, you will refactor this program

according to the instructions below,

and discuss the effect of the changes on performance and readability of the program.

%

Please do the following {\bfseries in exactly that order}.

%

{\bfseries Each step uses the result of the previous step.}

\begin{compactenum}

\item

Find a project partner (this assignment must be done in a 2-person team).

\item

Obtain `analyze.py' and data

(tiny English data, small German data, large Turkish data)

from the course website.

Run the program,

read the source code and understand how it works.

\item

Refactor the program as follows:

\bi

\item

Create a class NGramCounter(n)

that counts sequences of length n of words.

\item

Replace BigramCounter and TrigramCounter by NGramCounter.

\item

Check if the program works correctly and save it as

`analyzeRefactor1NGram.py'.

\ei

\item

\sloppy

Run `analyzeRefactor1NGram.py' on all three data sets

and observe the memory and time usage (for your report)

(memory is printed by the function `showMemTime').

\item

Reduce the memory usage by deleting big data when it is

no longer needed (see the `del' keyword).

Observe time and memory usage after this change.

Save that file as `analyzeRefactor2Delete.py'.

\item

Refactor the program so that you never load the complete file.

You can achieve this using the iterator interface:

\begin{verbatim}

f = open(filename,'r')

for line in f:

# do everything you need to do with 'line' here

# (for example counting bigrams/trigrams)

# (the line data will be deleted automatically by python)

\end{verbatim}

Observe memory and time usage after this change.

Hint: convert each line to unicode instead of the whole file.

Save that file as `analyzeRefactor3Iterator.py'.

\item

Refactor the program again:

\bi

\item

Create a function `def displayKMostFrequentNGramsInFile(k,n,filename)'

which opens the file, creates a n-gram counter class for m,

counts the n-grams, and then prints the k most frequent n-grams.

\item

In your main function, do only the following:

call that function 5 times as follows:

show the 30 most frequent 2-grams,

show the 20 most frequent 3-grams,

show the 15 most frequent 4-grams,

show the 15 most frequent 5-grams,

show the 15 most frequent 6-grams.

(This will read the whole file 5 times, this is ok.)

\item

Observe memory and time usage after this change.

Save that file as `analyzeRefactor4More.py'.

\ei

\item

Put the 4 programs you created so far into a .zip archive.

\item

Run the programs and fill in the empty cells of the following table:

\begin{tabular}{|l|c|c|c|c|c|c|c|}

\hline

program

& lines & \multicolumn{6}{c|}{resource usage} \\

& of code & \multicolumn{2}{c|}{English data}

& \multicolumn{2}{c|}{German data}

& \multicolumn{2}{c|}{Turkish data} \\

& & mem & time & mem & time & mem & time \\

& (\#) & (max MB) & (s) & (max MB) & (s) & (max MB) & (s) \\

\hline

analyze.py & & & & & & & \\

analyzeRefactor1NGram.py & & & & & & & \\

analyzeRefactor2Delete.py & & & & & & & \\

analyzeRefactor3Iterator.py & & & & & & & \\

analyzeRefactor4More.py & & & & & & & \\

\hline

\end{tabular}

\item

Write a report (1 page) where you describe:

\bi

\item

The time and memory usage of steps 3, 4, 5, 6, and the table from step 8.

\item

Comment on the lines of code, memory, and time usage of the programs.

\item

Is there any duplicate code left which could be eliminated?

\item

Which step above made the program more understandable/readable

and which made it less understandable/readable?

\item

What conclusions do you get from this assignment?

\ei

\item

Send your report (PDF) and your .zip archive to\newline

berna.altinel@gmail.com and in CC to peter.schuller@marmara.edu.tr .

Include both of your names and both email addresses in the email!

\end{compactenum}

\end{document}

% vim:ts=2:noet:sw=2: