Session 6 (14:00 – 17:00, Fri.): Miscellanea and personal interests

- Modifying existing scripts to make them work: considering the associated problems (and dangers) and looking at some examples.
- Your interests: one-to-one conversations with attendees to offer personalized suggestions and scripting challenges.
- Free practising session: practice personalized challenges and/or contents from any of the previous modules.

6.1. Modify existing scripts to make them work

6.1.1. Considering the problems

- I have good news and bad news. The good news is that modifying a script to make it work **is totally doable**. The bad news is that the feasibility of doing it is completely dependent on what the script is, what it does, how long it is, how well or horribly is written, and how much time you have to spend on the script.
  - There is no such a thing as "**the bits you're supposed to alter** to make the script work", unless the modifications are trivial (e.g., changing output tab-separated files to comma-separated files).
- Unless you're willing to take huge risks, the only way to be safe is to **read the script from top to bottom**, understand every single bit of it (line, by line), and once you really know what's going on, you can modify it to suit your needs.
- When scripts are long (e.g., more than 20 lines long), this process is highly time-consuming, particularly given how badly scripts are normally written, and how often they are recycled. The three typical solutions are:
  - The risky and bad one: **fiddling** with the script until it “works”. Most people do this, and consequently run scripts that don't understand or from unreliable sources and that might not be doing what they're supposed to, and then you get very bad science.
  - The slow and perhaps good one: **reading the whole thing**, repair its bugs, commenting it, updating syntax, and then modifying it to suit your needs.
  - The (potentially) slow and often times better one: **writing your own scripts**, using the best possible standards of scripting and updated syntax.

6.1.2. Looking at some examples

- Now we'll be taking a look to 5 scripts **inspired in some real examples** that you can find available online. Four of the original scripts were written by Mietta Lennes (http://www.helsinki.fi/~lennes/praat-scripts/) and one by Jonas Lindh (https://sites.google.com/site/phoneticolindh/home/scripts-and-audio/scripts).
  - I'd like to emphasize that the version of the scripts that you'll be seeing here is a **highly modified** version of the original scripts. All the scripts worked fine before I modified them, although they were using old Praat syntax and some risky scripting techniques (e.g., variable substitution).
- Our task will be to tidy up the scripts, to try to understand what they do, to identify their problems, to **fix all bugs**, to comment it and to update the syntax.

- **First example**:

```
############################################################################
# Modified by Mauricio Figueroa (www.mauriciofigueroa.cl) for teaching
# purposes. The script has been modified to contain several fatal bugs as well
# as other documentation and stylistic problems. This script should not be used
# for research purposes.
############################################################################

# This script will calculate the durations of all labeled segments in a
# TextGrid object. The results will be save in a text file, each line
# containing the label text and the duration of the corresponding segment.
# A TextGrid object needs to be selected in the Object list.
# This script is distributed under the GNU General Public License.
# Copyright 12.3.2002 Mietta Lennes

form Calculate durations of labeled segments                                    (6.1)
integer Tier 2
text textfile TEST_TEST.txt
endform
numberOfIntervals = Get number of intervals...: tier
for interval from 1 to numberOfIntervals
label$ = Get label of interval... tier, interval
if label$ <> ""
start = Get starting point... tier interval
end = Get end point... tier interval
duration = end - start
resultline$ = "'label$       'duration''newline$'"
fileappend "'textfile$'" resutline$'
endif
endfor
```

- **Second example**:

```
############################################################################
# Modified by Mauricio Figueroa (www.mauriciofigueroa.cl) for teaching
# purposes. The script has been modified to contain several fatal bugs as well
# as other documentation and stylistic problems. This script should not be used
# for research purposes.
############################################################################

# This script will draw an LPC spectrum from a given window around the cursor
# in the editor window.
# 11.3.2002. Mietta Lennes

form Draw an LPC spectrum from a small window around the cursor
comment LPC options:
integer Prediction_order 20                                                     (6.1)
positive Analysis_width_(seconds) 0.025
positive Time_step_(seconds) 0.005
positive Preemphasis_from_(Hz) 50.0
comment Options for drawing the spectrum from LPC:
integer Minimum_frequency_resolution_(Hz) 20
positive Bandwidth_reduction_(Hz) 0.1
positive Deemphasis_frequency_(Hz) 50.0
endform
cursor = Get cursor
start = cursor - analysis_width
end = cursor + analysis_width
Select... start end
milliseconds = round (cursor * 1000)
```

```
Extract windowed selection... LPC_'milliseconds'ms Kaiser2 2 no
endeditor
To LPC (burg)... prediction_order analysis_width time_step preemphasis_from
To Spectrum (slice)... analysis_width minimum_frequency_resolution
bandwidth_reduction deemphasis_frequency
Edit
editor Spectrum LPC_'milliseconds'ms
Zoom... 0 5000
endeditor
select Sound LPC_'milliseconds'ms
Remove
select LPC LPC_'milliseconds'ms
Remove
editor
Move cursor to... cursor
```

- **Third example**:

```
############################################################################
# Modified by Mauricio Figueroa (www.mauriciofigueroa.cl) for teaching
# purposes. The script has been modified to contain several fatal bugs as well
# as other documentation and stylistic problems. This script should not be used
# for research purposes.
############################################################################

# Jonas Lindh 2006
# http://www.ling.gu.se/~jonas

form Helium speech
comment choose recording duration
positive Rec_dr(sec) 4
comment
comment Your name
text name Jonas
endform
echo Speak for 'Rec_dur' seconds 'name$'
Record Sound (fixed time)... Microphone 0.99 0.5 44100 Rec_dur
Rename... 'name'
Change gender... 75 600 1.2 0 1, 1
Rename... helium_'name$'
Play
echo You are heliumized name$
```

(6.1)

- **Fourth example**:

```
############################################################################
# Modified by Mauricio Figueroa (www.mauriciofigueroa.cl) for teaching
# purposes. The script has been modified to contain several fatal bugs as well
# as other documentation and stylistic problems. This script should not be used
# for research purposes.
############################################################################

# This script will open all the files in a given folder. All the files must be
# recognized by Praat (either sound files such as AIFF or WAV or Praat
# analysis files like TextGrid).
# This script is distributed under the GNU General Public License.
# Copyright 11.3.2002 Mietta Lennes

form Open all files in directory
sentence Directory ../files_to_open/
```

(6.1)

```
endform
Create Strings as file list... list 'directory$'*
numberOfFiles = Get number of strings
for ifile to numberOfFiles
filename$ = Get string... ifile
if right$ (filename$, 4) != ".wav"
Read from file... 'directory$''filename$'
endif
select Strings list
endfor
select Strings list
# Remove
```

- **Fifth example**:

(6.1)

```
#############################################################################
# Modified by Mauricio Figueroa (www.mauriciofigueroa.cl) for teaching
# purposes. The script has been modified to contain several fatal bugs as well #
as other documentation and stylistic problems. This script should not be used
# for research purposes.
#############################################################################

# This script calculates the total duration of those intervals in the selected
# tier which have a regular label or transcription (not those labeled 'xxx' or
# starting with).
# You can also give the name of a criterion tier and a criterion label: only
# those segments will be counted that are part of an interval in the criterion
# tier that has the criterion label.
# A TextGrid object has to be selected before running this script. [...]
# This script is distributed under the GNU General Public Licence.
# Copyright 19.3.2002 Mietta Lennes.

form Calculate the total duration of intervals
comment Calculate total duration of intervals in tier:
integer Duration_tier 2
comment Intervals labeled with 'xxx', starting with '.' or empty intervals will
not be included.
comment Additional criterion for included intervals: They must be part of
intervals in tier number
integer Criterion_tier 2
comment that are labeled as:
sentence Label The
endform
total_duration = 0
count = 0
numberOfIntervals = Get number of intervals... duration_tier
for i from numberOfIntervals
label1$ = Get label of interval duration_tier i
if label1$ <> "" and left$(label1$,1) <> "." and left(label1$,3) <> "xxx"
start1 = Get starting point... duration_tier i
end1 = Get end point... duration_tier i
duration = end1 - start1
middle1 = (start1 + end1) / 2
if criterion_tier < 0
criterion = Get interval at time... criterion_tier middle1
start2 = Get starting point... criterion_tier criterion
end2 = Get end point... criterion_tier criterion
label2$ = Get label of interval... criterion_tier criterion
if start2 <= start1 and end2 >= end1 and label2$ != lable$
total_duration = total_duration + duration
count = count + 1
endif
```

```
else
total_duration = total_duration + duration
count = count
endif
endif
endfor
# Print the results to the Info window
echo The total duration of the intervals in tier number 'duration_tier' was
calculated.
printline Those intervals whose label is "xxx" or starts with "." (dot) were not
included.
if criterion_tier > 0
printline Only those intervals that are part of another interval in tier
'criterion_tier'
printline having the label "'label$'" were included.
endif
duration_in_minutes = 'total_duration' / 60
printline
printline Total duration of the 'count' intervals (fulfilling the criteria) is
'total_duration' seconds.
printline That is 'duration_in_minutes' minutes.
```

6.2. Your interests and free practising session

- One-to-one conversations with attendees to offer personalized suggestions and scripting challenges.
- Practice personalized challenges and/or contents from any of the previous modules.