MODULE 6:
HUMAN FACTORS & ENGINEERING PSYCHOLOGY

Pair programming project

Course code: 201400123

2017-2018

Contact and Coordinator

Dr. Martin Schmettow

Location: Cubicus B324

email: m.schmettow@utwente.nl

1. PROJECT ASSIGNMENTS

In the following, three choices are described for the PPP. All three programs implement assessment procedures (AP) for aspects of human performance, relevant to the domain(s) of Human factors & Engineering Psychology. Every project description is structured as follows:

1. goal of the procedure: for what purpose the AP is used
2. description of the AP: outline of steps and used material
3. required functionality: what implemented functionality is minimally required for passing PPP
4. challenges: which programming concepts are particularly relevant and what are the tricky parts of the problem?
5. Difficulty score: which level of skills are required to create this program (more difficult programs give better grades, *if required functionality is implemented*)

Regarding functionality and challenges (3. – 4.), there is a common set for all the programs, see *Assessment criteria*.

1.1 SEMANTIC STROOP

Sometimes, researchers are interested in what people associate with certain objects. The following experimental paradigm uses a variant of the Stroop task to read the minds of participants.

Take, for example, the study by Schmettow, Noordzij, & Mundt (2013), which explored whether computer science students have different associations when seeing pictures of computers. The authors compiled three target words categories in conjunction with pictures showing computers in various situations. In the semantic Stroop task, the picture is shown before the word, to which the participant has to respond. When the participant experiences a strong association between picture and subsequent word, this causes a moment of distraction, resulting in longer response times.

In this project, you implement the semantic Stroop task. You use at least two word categories that are compared. It is recommended that you think of a research question to answer with your experiment. You may re-implement the original experiments of Schmettow et al. (2013) or (Sparrow, Liu, & Wegner, 2011), or think of other examples, such as:

- when seeing a smart phone, do people rather think of social themes (connecting with people, communicating, dating) or of finding information and getting their personal tasks done (reminding, shopping, weather forecast).
- who knows red riding hood? assessing whether a person knows classic fairy tales

1.1.1 REQUIRED FUNCTIONALITY

- it is possible to use the Stroop program as a template
- use a set of word categories
- displaying the stimuli correctly (picture → word)
- prepare stimuli in one of two ways:
    - reading a prepared table of picture – word pairs
    - create random picture – word pairs
- record reaction times of correctly answered and calculate one average response time per experimental condition (word categories)

- write a data file with three variables: |word category|correct|RT|

1.1.2 CHALLENGES

- read picture files and display them with PyGame commands

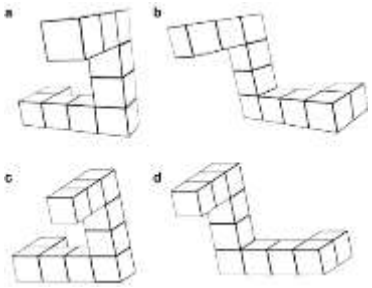
1.1.3 DIFFICULTY: EASY

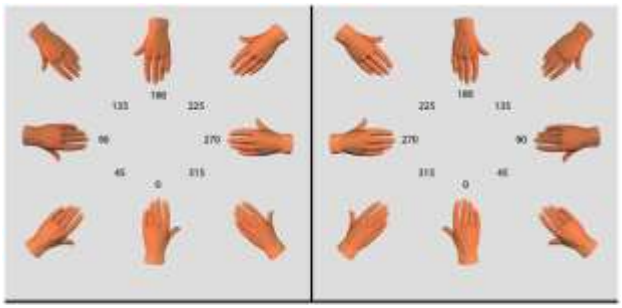
Correct and complete implementation of required functionality will typically give a 6. Using advanced programming concept (e.g., object orientation) or adding relevant and non-trivial functionality can give higher grades.

1.2 MENTAL ROTATION TASK

In many critical domains, operators perform tasks which require visio-spatial abilities. For example, surgeons must be able to imagine the location of organs and vessels or planning their actions.

One frequently experiment to assess visio-spatial abilities of a person, are mental rotation tasks. Participants are shown pairs of objects and they must decide whether these objects are rotated versions of the same shape.





For answering this question, participants must perform a mental rotation of the displayed objects. The reaction time is measured as an indicator for a person's ability. Furthermore, response times increase linearly with greater angular disparity between objects.

You implement the mental rotation task using a set of stimuli, which you either

- produce yourself (e.g., taking pictures of rotated hands)
- create inside the program (e.g., rotated and inflected letters)
- or download somewhere else

See http://plato.stanford.edu/entries/mental-imagery/mental-rotation.html for some variants of stimuli, or Parsons (1987).

1.2.1 REQUIRED FUNCTIONALITY

- displaying the stimuli correctly
  - reading a prepared table of object pairs
  - or create random picture pairs
- record reaction times of correctly answered and calculate one average response time per experimental condition (match, non-match)
- record the disparity angle as covariate
- write a data file with four variables: |mirrored|correct|angle|RT|

1.2.2 CHALLENGES

- read picture files and display them with PyGame commands

1.2.3 DIFFICULTY: MODERATE

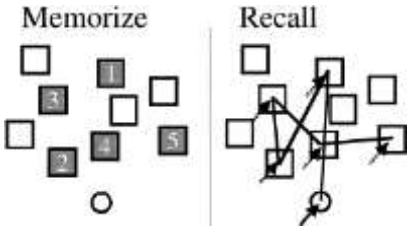
Correct and complete implementation of required functionality will typically give a 7. Using advanced programming concept (e.g., object orientation) or adding relevant and non-trivial functionality can result in higher grades.

1.3 CORSI BLOCK TAPPING TASK

The working memory (WM) plays a crucial role in all kinds of decision, problem solving and wayfinding tasks. For example, Freudenthal (2001) found that elderly people are slower at information retrieval tasks, because of decline in working memory capacity (WMC).

All theories on WM assume that the WM has a person-specific capacity or *span*. The earliest example is the famous Miller's magic number of 7±2, which has been suggested as a rule for web design (Larson & Czerwinski, 1998). Modern theories assume that the working memory consists of separate components for visual-spatial and auditory information.

The Corsi block tapping task is a test a person's visual WMC (Berch, Krikorian, & Huha, 1998). It shows an unordered pattern of blocks. A sequence of particular length is shown to the participant. Then, the participant is asked to reproduce the sequence. If a participant fails two times on a sequence of length S then the visio-spatial WMC is S-1.



1.3.1 REQUIRED FUNCTIONALITY

- drawing one of the standard block patterns to the screen (Berch et al., 1998)
- displaying sequences of various length by
  - reading in prepared sequences
  - or randomly creating the sequences
- recording user input using mouse events
- checking user input for correctness and compute the final score

1.3.2 CHALLENGES

- a rather abstract procedure is needed for timing the sequence
- interaction using mouse clicks
- recording and comparing the sequence is not trivial

1.3.3 DIFFICULTY: HIGH

Correct and complete implementation of required functionality will typically give an 8. Using advanced programming concept (e.g., object orientation) or adding relevant and non-trivial functionality can result in higher grades.

2. ASSESSMENT CRITERIA

All projects are graded on four criteria:

- difficulty level (see above)
- functionality
- code quality
- documentation

Every project is first checked for the minimum criteria on functionality, code quality and documentation:

Minimum criteria, all programs must fulfil:		
functionality	code quality	documentation
❒ have a welcome screen ❒ have a sequential presentation of trials ❒ have a goodbye screen ❒ present stimuli graphically, using PyGame commands ❒ feedback after every trial ❒ be interactive, using key presses, at least. ❒ record participant responses ❒ print summary statistics at the end of the experiment, at least ❒ program-specific required functionality (see descriptions above)	❒ string and numerical variables ❒ logical and arithmetic operators ❒ lists ❒ conditionals ❒ for loops ❒ PyGame events ❒ basic PyGame drawing functions	❒ a complete state chart ❒ sufficient comments in the code

If minimum requirements are just met, the project will receive the grade matching its level of difficulty. project receive better grades if they go beyond the required criteria in functionality, code quality and documentation:

Examples of additional criteria, that give better grades:		
functionality	code quality	documentation
❒ more complex flow, e.g., automatic transitions, branching ❒ more complex graphical elements, such as buttons to click ❒ graphical presentation of results, e.g. boxplots ❒ flexibility and versatility of the program, e.g. random generation of stimuli ❒ eye candy, such as animated instructions	❒ using two-dimensional arrays ❒ using dictionaries ❒ using additional Python libraries ❒ using advanced drawing commands, such as transformations or sprites ❒ creating parameterized functions ❒ creating own classes (object orientation) ❒ elegant algorithms, such as updating a set of objects using lists and loops	❒ extensive commenting ❒ sketches of screen layout

3. HANDING IT IN

Like all other assignments, projects are handed in via Blackboard. Instructions for submission:

1. Configure your program that I can test it quickly: it suffices to run three trials per category. Use short presentation times for the images (1s).
2. Keep all required files (.py, .png, .jpg, .wav etc.) in one directory. Avoid sub folders and never put required files in a completely separate folder.
3. Only if you know how to use relative paths, should you use sub folders.
4. If you use additional libraries that need to be installed, list them in a README file (text or word file)
5. If running your program requires any other preparations, describe that in the README (but better avoid that altogether)
6. Create a zip-file (e.g. with 7-Zip) or rar-file (Apple) of the directory with source files
7. Do not submit additional information on the submission page of Blackboard. I could miss it, because I will just download all zip files in a batch.
8. Highly recommended: before submitting the zip or rar file, send it to another student (who is not in your group). Can this person just unzip and run the program?

REFERENCES

Berch, D. B., Krikorian, R., & Huha, E. M. (1998). The Corsi block-tapping task: methodological and theoretical considerations. *Brain and Cognition*, *38*(3), 317–38. doi:10.1006/brcg.1998.1039

Freudenthal, D. (2001). Age differences in the performance of information retrieval tasks. *Behaviour & Information Technology*, *20*(1), 9–22. doi:10.1080/0144929011004974

Larson, K., & Czerwinski, M. (1998). *Web page design: implications of memory, structure and scent for information retrieval. Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '98* (pp. 25–32). New York, New York, USA: ACM Press. doi:10.1145/274644.274649

Parsons, L. M. (1987). Imagined spatial transformations of one's hands and feet. *Cognitive Psychology*, *19*(2), 178–241. doi:10.1016/0010-0285(87)90011-9

Schmettow, M., Noordzij, M. L., & Mundt, M. (2013). An implicit test of UX: Individuals Differ in What They Associate with Computers. In *CHI '13 Extended Abstracts on Human Factors in Computing Systems on - CHI EA '13* (pp. 2039 – 2048). New York, New York, USA: ACM Press. doi:10.1145/2468356.2468722

Sparrow, B., Liu, J., & Wegner, D. M. (2011). Google effects on memory: cognitive consequences of having information at our fingertips. *Science (New York, N.Y.)*, *333*(6043), 776–8. doi:10.1126/science.1207745