PGG311T 2019 Semester 2
Assignment 1: Marks up for Grabs

To succeed in life opportunities must be secured. Grab your "marks" fast and a "pass" is guaranteed! This little game simulates your study challenge. The quicker you go for the marks the higher your score and the better the chance to build up a good average. A provided demo application will demonstrate the activities.

The grabbing of marks do start with a click on the *bitmap* button "Go!".

Two square sized panels (40 – 60 pixels a side) pop up at random positions. One green and one light blue. The caption on each respective panel indicates the possible mark that can be scored if you click on it at the very same split of a second. The starting values for all panels should be any random number from 80% – 100% *inclusively*. Unfortunately good marks don't stay for grabs for ever. Each of those marks/ panels start "shrinking" or "falling" immedeately after their re-positioning.

The "Disappearing Speed" (interval) in our example "100 ms" indicates the timelaps till the next shrinking or "falling" is happening. With each (interval) cycle one panel will *shrink* but stay at its position, the second panel is slowly "*dropping*" as it is shrinking. See demo and figure 1.

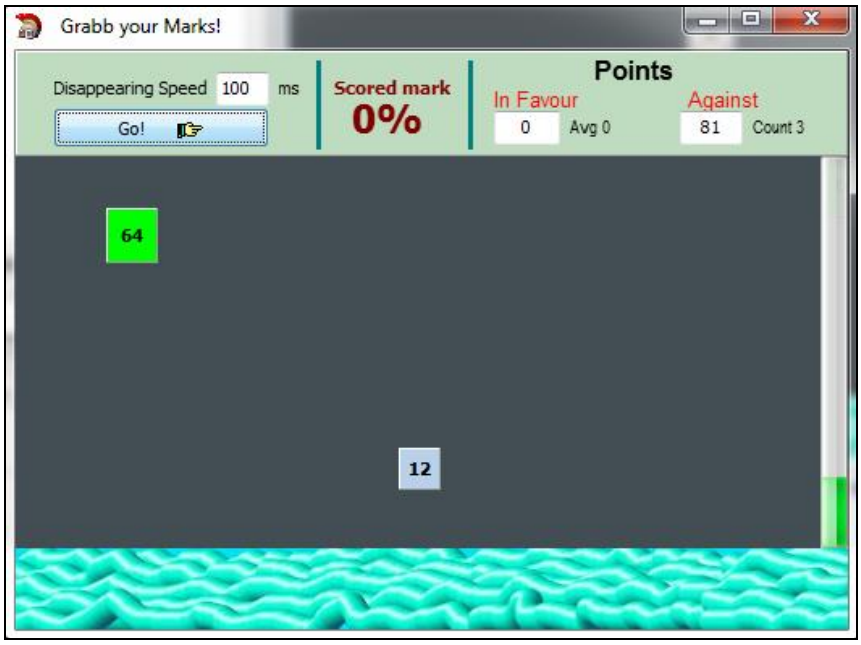


Fig 1: Start of application

The progressbar on the right indicates the duration of the game. It finishes after 30 seconds.

After the game is over the user gets a choice via a message dialog to play another time (resetting all values to initial values) or quitting the game. If the game is quitted a message dialog confirms the quitting.

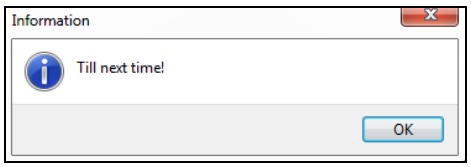


Fig 2: Terminating the game with a message

Even after resizing the form the mark panels should still fit inside the allowable space. See fig 3.

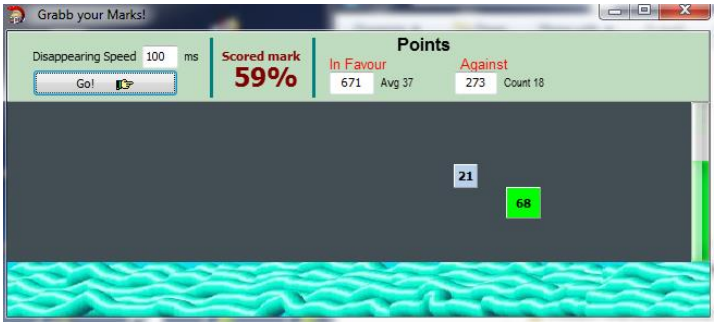


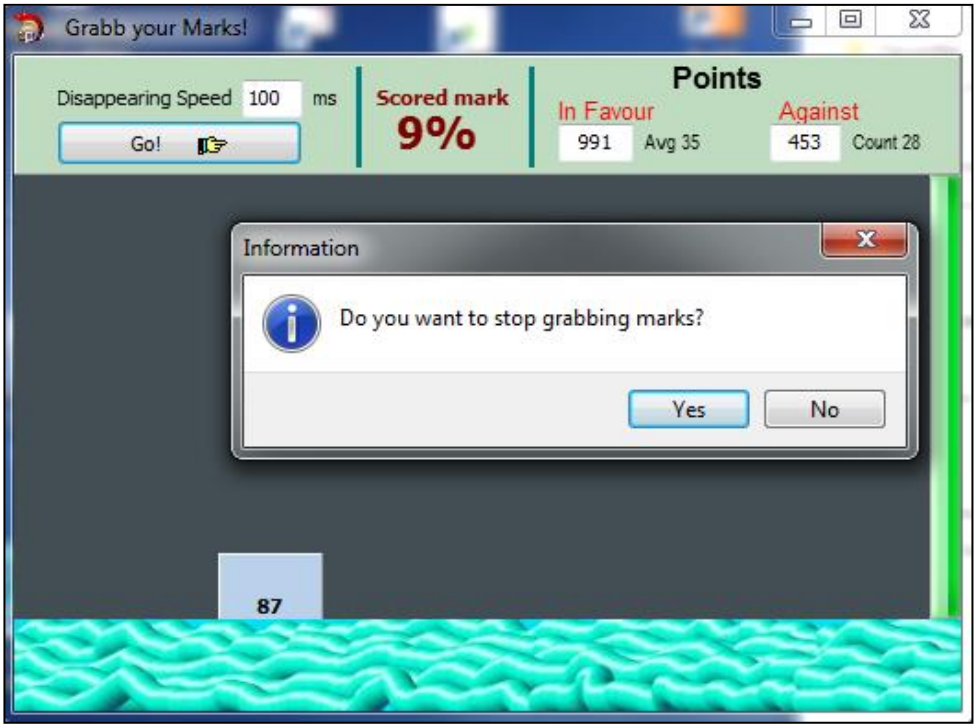
Fig 3: The form was re-sized to be wider

The header part needs some explainations (see fig 04):

- The cycle for the timer (interval) is currently set to 100 ms. Allowable range: 50 – 200!
- The last panel the user clicked on, gained him/her a score of 53%.
- All ***scored marks*** from any clicked on "panel" will be added up to produce the "**In Favour**" number. Our example shows 1087.
- All "panels" you never reached (ie you didn't click on it) will be added up with ***its starting value*** to produce the "**Against**" points of 879 (in our example). For a good quick player this number shouldn't grow too big.
- The ***number of panels*** that appeared during the execution is summarized under "Count". You can see "Count 40" in our example.
- The "Avg 27" is the calculated average of all scored marks (1087) divided by the total count (40). This 27% is actually your true result you achieved overall.

Fig 4: Header with typical numbers.

Fig 5: Mark grabbing comes to an end



~~ Enjoy the challenge of problem solving and programming. ~~

[On the next page the handing in procedure is explained]

Appendix A

- **Assignments** (Compare with Study Guide)

  Two to four assignments will be issued to be completed. The assignments are based on the skills to be acquired and will support the understanding of the learning content. It is critical important to complete these assignments ***on your own*** to improve your programming skills and understanding of the programming content. Please refer to the work schedule for issue dates and deadlines. All assignments and notifications will be provided as part of ***myTUTOR.***

  It is allowed to consult lecturers, TAMP students, other students, etc – but ***copied assignments*** from other students or any other illegal source will gain you 0%.

Assignments are handed in *during class time* in your respective group through "Send to Lecturer". Each assignment handed in must have the following **TWO** components in a well format way:

1. A ***PDF file*** with student number and assignment number **as file name**,

eg: **210123123_ Assignment_1.PDF**

A. **Cover page** stating student-number, subject-code, semester and assignment number
B. **Screen prints** to provide proof of a good running program. All-important moments of your running program should be fixed and documented in screen prints. The quality and efficiency with meaningful captions will contribute to a higher mark.
C. **Program source code**. A print out of important source code lines. All code must be printed in mono-spaced font (eg: *New Courier*). Automatically inserted line breaks by the printer are unacceptable (make sure your lines do not exceed 80 columns). Single line spacing is preferred.

PDF File

[This PDF file will be marked!]

2. A ***ZIP file*** with student number and assignment number and the word "CODE".

D. All programs written and submitted should include *descriptive component names* as well as *comments – including your name and student number.*
E. The complete source code (*cleaned*) must be zipped and named as follows:

ZIP File

210123123_Assign_1 CODE.ZIP

[Your zip-file must be **smaller than 10 MB**!]

Late assignments (and/ or emails) are not accepted ;(