Exercise 9 (10 points) – can be done in pair or individually

- The first lines of all source files must be comments containing names & IDs of all members. Also create file readme.txt containing names & IDs of all members

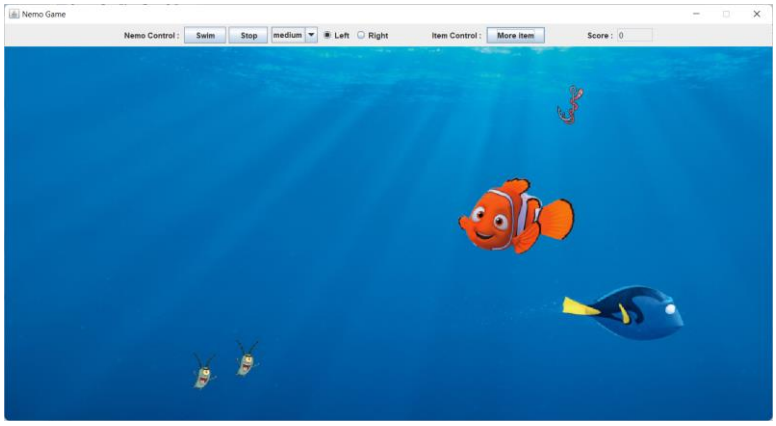
- Put all files (source, input, readme.txt) in folder Ex9_xxx where xxx = ID of the group representative, i.e. your source files must be in package Ex9_xxx (assumedly in Maven's src/main/java). Input files must be read from this path

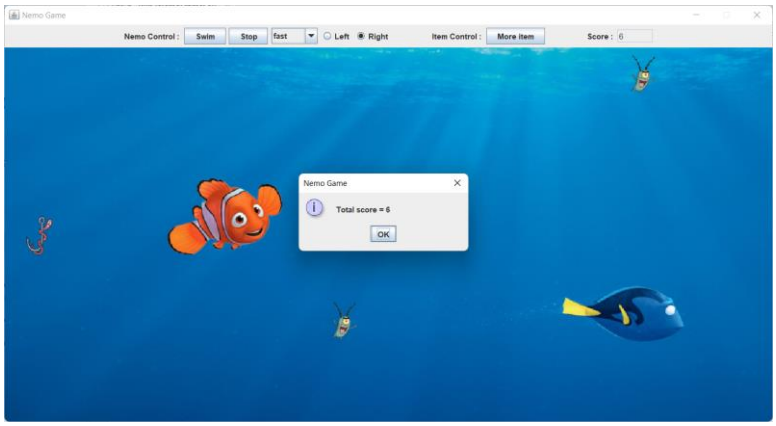
- The group representative zips Ex9_xxx & submits it to Google Classroom. The other members submit only readme.txt. Email submission is not accepted

==========

Use the given image/sound files and source file (NemoFrame.java). Folder resources must be put inside your project folder (Ex9_xxx). Complete the source file to make program work as follows:

Nemo and items (hooks and planktons) are controlled by separate threads





1. Nemo control
   1.1 Swim/Stop buttons: swim/stop
   1.2 Combo box: set Nemo's speed
   1.3 Radio buttons: turn left & walk to the left, or turn right & walk to the right. When reaching one side of the frame, he'll appear on the other side

2. Item control
   2.1 More button: random 1 item which can be hook or plankton
   2.2 Hook starts at random X at the top and falls down
   2.3 Plankton starts at random X at the bottom and floats up
   2.4 Update score when hook/plankton hits Nemo

3. Report total score when closing frame

4. All listener classes must be anonymous classes. Add listeners as follows

4.1 Add ActionListener to Swim & Stop buttons, to make Nemo swim or stop

- Swim → create and start nemoThread
- Stop → stop nemoThread

4.2 Add ItemListener to combo box, to set Nemo's speed

- Fast = short sleeping time for nemoThread
- Slow = long sleeping time for nemoThread

4.3 Add ItemListener to each radio button, to set Nemo's direction

4.4 Add ActionListener to More button, to add a random item. It can be done by creating & starting a new itemThread (each item is controlled by each thread)

4.5 Add WindowListener to the frame, to show the final score when closing it

5. Use nemoThread & itemThread to make all labels move automatically. Anonymous class can also be applied. Complete method setItemThread and class ItemLabel, using example from setNemoThread and NemoLabel

6. Complete method updateScore to increase/decrease score when an item hits Nemo. This method requires proper synchronization because it can be called by multiple itemThreads

All given code can be modified as needed