Exercise 8 (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 Ex8_xxx where xxx = ID of the group representative. That is, your source files must be in package Ex8_xxx and input files must be read from this path

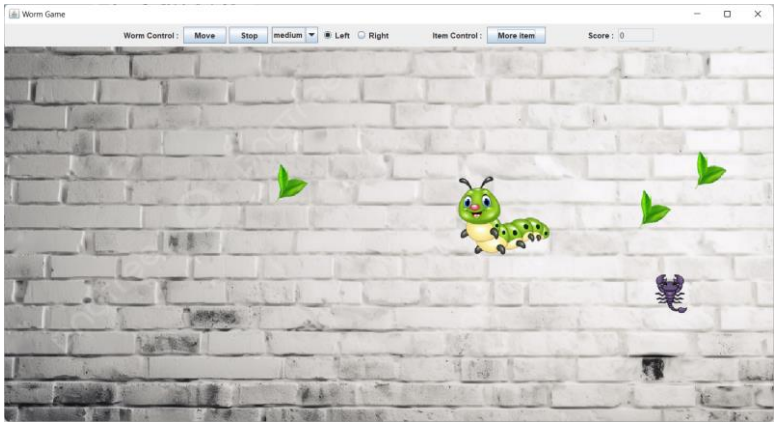
- The group representative zips Ex8_xxx & submits it to Google Classroom. The other members submit only readme.txt. Email submission is not accepted

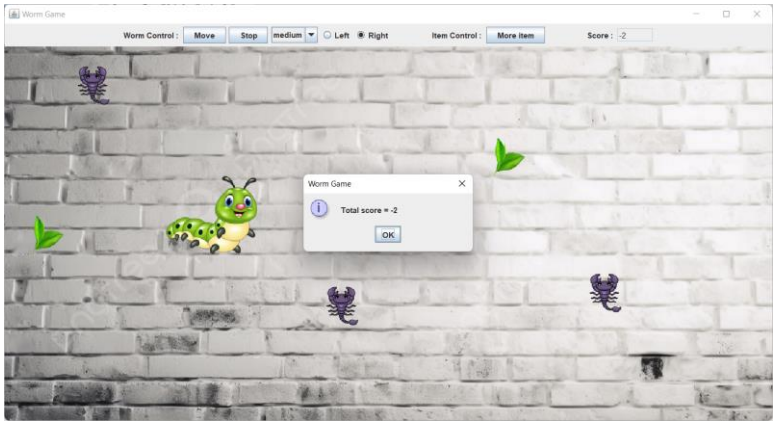
==========

Use the given image/sound files and source file (MainApplication.java). Unzip resources.zip and put this folder in your project folder (Ex8_xxx)

Complete the source file to make program work as follows:

Worm and items (leaves and scorpions) are controlled by separate threads





1. Worm control
   1.1 Move & Stop buttons to move & stop
   1.2 Combo box to set Worm's speed
   1.3 Radio buttons to turn left & move to the left, or turn right & move to the right. When reaching one side of the frame, it'll appear on the other side
2. Item control
   2.1 Item button to random 1 falling item which can be leaf or scorpion
   2.2 Leaf starts at random X at the top and falls down
   2.3 Scorpion starts at random X at the bottom and floats up
   2.4 Update score when Leaf/Scorpion hits Worm
3. Report total score when closing frame

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

4.1 Add ActionListener to Move & Stop buttons, to make Worm move or stop

- Move → create and start wormThread
- Stop → stop wormThread

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

- Fast = short sleeping time for wormhread
- Slow = long sleeping time for wormThread

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

4.4 Add ActionListener to Item 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 wormThread & itemThread to make all labels move automatically. Anonymous class can also be applied. Complete method setItemThread and class ItemLabel, using example from setWormThread and WormLabel

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

All given code can be modified as needed