IDMT 2020

Assignment 3

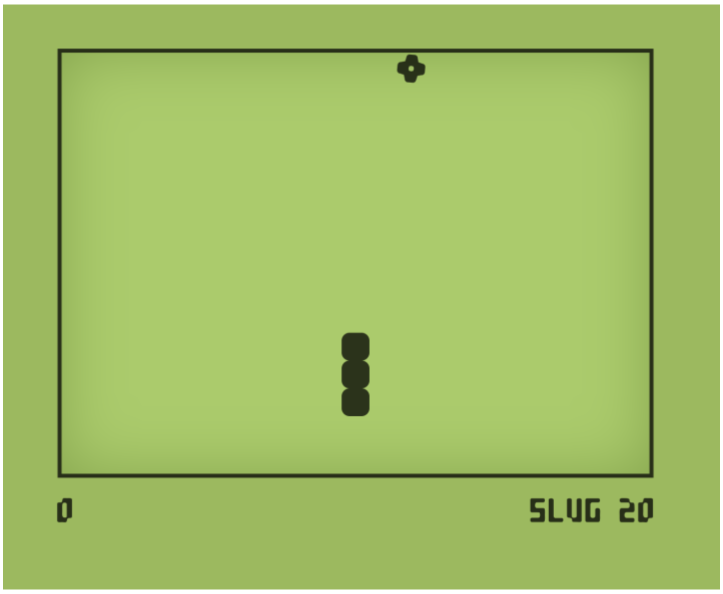
(Version: 20201102)

This assessment is 20% of your total IDMT module grade.

The purpose of this assignment is to help you become familiar with using Pure Data to create digital audio and multimedia interfaces. You will use Pure Data to add sonic interaction elements to the simple snake game you created using Processing and Arduino in Assignments 1 and 2.

Recap: How the Snake game is played

Example game:
https://playsnake.org
(Keys for this example version are: left: A; right: D; up: W; down: S)



Snake starts at 3 units long and is always moving forward.
Use Arduino buttons to change direction of snake: left, right, up, down.
Snake dies when it hits the wall (or runs into itself).
Each time the snake eats an apple, it grows by 1 unit and the points increase.

Instructions

You should write 5 triplets of Pd, Arduino, and Processing patch and sketches – one triplet for each of the steps below. Each step builds on the previous one, so make copies of your code as you move from one step to the next. Each sketch should be self contained, be runnable without modification, and include all the functionality of the previous steps as well as the current step.

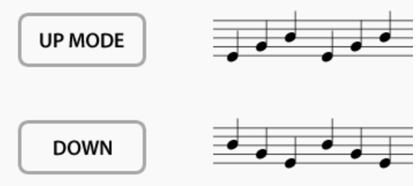
Note 1: The description for each step below focuses on the required Pd outcome. Controls with sensors via Arduino should be as in Assignment 2, but Arduino sketches might need some modifications in case they communicate directly with Pd. Processing sketches should also be as in Assignment 2, but with necessary modifications to communicate with Pd, as well as one new requirement.

Note 2: You need to improve your Assignment 2 code to successfully complete Assignment 3. The teaching team and demonstrator can help you with that.

Step 1: Each unit of the snake is one note from a 3-note arpeggio (patch/sketches name: Step1)

Step 2: Control the filter and transposition of the notes in your arpeggio with the direction of the snake (patch/sketches name: Step2):

- **(Step 2a)** Each time the snake turns up/down the notes are transposed up/down by 1 octave (i.e. 2*frequency or 0.5*frequency)
- **(Step 2b)** Each time the snake turns left/right the direction of the arpeggio changes to up/down.



Step 3: Control the speed of the arpeggio with the speed of the snake (patch/sketches name: Step3)

Step 4: Make a "crash" or "endgame" sound (i.e. trigger a stored sound file) when the snake hits a boundary (patch/sketches name: Step4):

Step 5: Each time the snake eats an apple (patch/sketches name: Step5):

- **(Step 5a)** Pause the arpeggio and play a very short sound (as in the example above)
- **(Step 5b)** Make the snake grow by a unit (as in the example above)
- **(Step 5c)** Add a new note to your arpeggio and restart it.
- For example, when the snake eats its first apple, it becomes a 4-unit snake and a 4-note arpeggio.

Marking Scheme

You receive the following % of your overall IDMT module grade.

Step	Sketch pairs	Description
Step 1	0%: Not working at all 1%: Partially working 2%: Fully working	0%: No description or a list of comments from the code 1%: Unclear description 2%: Clear description
Step 2a	0%: Not working at all 0.5%: Partially working 1%: Fully working	0%: No description or a list of comments from the code 0.5%: Unclear description 1%: Clear description
Step 2b	0%: Not working at all 0.5%: Partially working 1%: Fully working	0%: No description or a list of comments from the code 0.5%: Unclear description 1%: Clear description
Step 3	0%: Not working at all 1%: Partially working 2%: Fully working	0%: No description or a list of comments from the code 1%: Unclear description 2%: Clear description
Step 4	0%: Not working at all 1%: Partially working 2%: Fully working	0%: No description or a list of comments from the code 1%: Unclear description 2%: Clear description
Step 5a	0%: Not working at all 0.5%: Partially working 1%: Fully working	0%: No description or a list of comments from the code 0.5%: Unclear description 1%: Clear description
Step 5b	0%: Not working at all 0.5%: Partially working 1%: Fully working	0%: No description or a list of comments from the code 0.5%: Unclear description 1%: Clear description
Step 5c	0%: Not working at all 0.5%: Partially working 1%: Fully working	0%: No description or a list of comments from the code 0.5%: Unclear description 1%: Clear description

Note that this scheme awards 2 bonus marks, i.e. a max of 22% of your overall IDMT grade.

Submission

Your submission should consist of the following:

• 5 Pd, 5 Arduino, and 5 Processing folders containing the code and any additional materials (e.g. audio files) for each step above. Your code should include clear comments to explain what each line of the code does. Make sure to credit any code that you have used in your sketches, *including any help from the teaching team or demonstrator in improving your sketches from Assignment 2.*

• Short report (PDF, max 2 sides of A4) describing how your code works for each step above as well as a circuit diagram of the circuit connected to Arduino (same as in Assignment 2 or improved based on feedback received). There should be an explanation for each step – explain what the important variables are, why you use those variables, and how the code achieves the objectives of each step i.e. how it works and why you wrote it that way. Explain how the circuit works and why you made it that way. Make sure to credit any code that you have used in your patch and/or sketches and explain how your code is different. You should include a section on your reflections about your code and how it works. This report should be written in plain English and should not just be a summary of the comments in your code. This report should emphasise the Pd implementation and summarise any decisions related to Assignment 2 – i.e. do not replicate most of your report from Assignment 2, but instead focus on how you have integrated sonic interactions via Pd.

• Name your files with your initials and surname and student number

• Make a .zip or .tar.gz file of all your code and report and submit it via QMplus. *Note that QMplus has a limit of 50 MB per submission.*

• Make sure the filename of the file you submit to QMplus includes your initials, surname, and student number

Remember to include your name and QMUL email in both your code and report