Minisoap

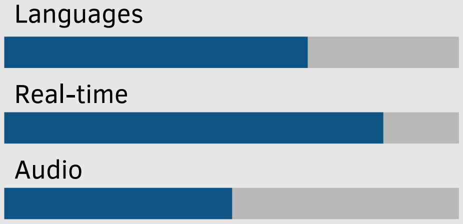
A modular audio processor

3/4 people

Project Description

An audio stream processor that can create and transform audio streams, featuring basic synthesis capabilities as well as file and soundcard input/output. It is designed modularly, allowing the user to specify its own processing pipeline.

Skills



(*)[The skill scale is from 0 (Fundamental Awareness) to 6 (Expert).]

Level 1 You may now pursue to the level 1 of the project.

⋆⋆⋆ Stream processors
Design a notion of (audio) stream generator that works with buffers (not samples) for efficiency, and has a notion of track. A stream may become unavailable at the end of a track. It may become available again after some delay.

⋆ Basic generators
Implement a silence generator, and a sine generator parameterized by its frequency and amplitude.

⋆ Scheduling operators
Implement fallback and rotation operators. Both take a list of input streams. The first one repeatedly plays a (complete) track from the first available source. The second one plays a track from an input, and then the next one from the next available input (rotating back to the beginning of the list if needed.)

⋆ Mixing
Implement an operator that mixes its input.

⋆⋆ Input/Output
Implement a source that reads from the soundcard, and make it possible to output a stream processor on the soundcard.

⋆⋆ Files
Implement input/output from/to one common audio file format.

⋆⋆ Playlists
Implement a stream generator which, given a playlist file, plays its files sequentially. It should offer shuffling and repeat options.

⋆⋆ Transitions
Implement transitions between tracks: the stream should be modified at the end of a track and at the beginning of the next one. At least fading should be supported.

⋆⋆⋆ Language
Make it possible for users to describe an audio processing pipeline in a simple domain-specific language.

Level 2 Level 1 must be unlocked to read this section

Paint is not dead

Collaborative drawing

4 people

Project Description

This software enables the user to draw pictures on its computer. It can be freehand drawing, but the drawing may also be guided by the computer in order to draw specific shapes (circles, squares...). It can contain more advanced features, with for example edge detections and automatic color filling.

Skills

Image manipulations

GUI

(*)[The skill scale is from 0 (Fundamental Awareness) to 6 (Expert).]

Level 1 You may now pursue to the level 1 of the project.

⋆⋆	GUI A user interface allows to create a new drawing of specified size, select the drawing tool, and display the drawing.
⋆	Basic tools It should be possible to perform freehand drawing, and draw basic shapes such as squares, circles, etc.
⋆	Parameters When drawing shapes or lines, different options are available, such as thickness, color, filling,...
⋆	Save The drawing can be saved and loaded.
⋆	Previsualization In the GUI, tools assist the creation of shapes with a live previsualization.
⋆⋆	History A drawing is equipped with a history of modifications. Undo/redo operations are available.
⋆⋆⋆	Team Effort In the spirit of etherpad, a server hosts each drawing, and several people can edit at the same time a drawing. The server must not make any rendering. Modifications in the history are tagged with their author.

Level 2 Level 1 must be unlocked to read this section

Dungeon Battle

A rogue-like game

4 people

Project Description

Dungeon crawls, or more specifically rogue-likes are turn-based computer games in which a single player evolves through a procedurally generated dungeon, fighting creatures, finding various objects, acquiring experience. The final objective may be to reach the top (or bottom, if the dungeon is a cave) of the dungeon and come back, or to find a special object. Death is typically permanent, and plays can be very short, especially for beginners.

Skills

Scenario

Real Time programming

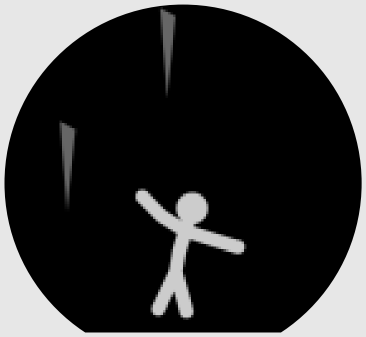
GUI

(*)[The skill scale is from 0 (Fundamental Awareness) to 6 (Expert).]

Level 1 You may now pursue to the level 1 of the project.

⋆	Board The game is played on a series of dungeon levels which are simple square grids. The grid cells may be empty, walls, or floor. Empty cells must never be next to floor cells. Maps can be loaded from and save to a file.
⋆⋆⋆	Graphics The graphics can either be 2D or text based. There is a window where the board is displayed.
⋆	Controls The player is controlled with zqsd and can move through floor cells, but should be blocked by walls.
⋆	Monsters Monsters appear on the map, and move along some predefined path.
⋆⋆	Fighting The hero can attack monsters, and kill them.
⋆	Experience The hero can progress and level up doing more and more damages the more monsters he kills.
⋆⋆	Intelligent Monsters Monsters will go towards the hero, without getting stuck. In simple situations without other moving entities, an optimal path is used.

Level 2 Level 1 must be unlocked to read this section



Dodge man

A one screen game

3 people

Project Description

The player is stuck inside the window, and objects are falling from the sky. If the player is touched, he dies, and if a dodges an object, the object disappears after touching the ground. The longest the player stays alive, the highest the score, but the hardest it gets.

Skills

GUI

Real time programming

(*)[The skill scale is from 0 (Fundamental Awareness) to 6 (Expert).]

Level 1 You may now pursue to the level 1 of the project.

⋆⋆ Window
A character, a background, and objects can be rendered.

⋆⋆ Objects
Objects are generated in semi random fashion, and are falling vertically to the ground.

⋆⋆ Character physics
The character can move left or right, jump and crouch. Details (e.g. double-jump) are up to game designers but must be documented and tested. The player cannot move through walls and floors.

⋆ Death
When hit, the character dies.

⋆ Score
There is a score system, with high score and leaderboard.

⋆⋆ Gameplay
The more the player progresses, the harder the game gets. This implies having the objects falling in a smart way which can support the acceleration. There should always be a winning strategy

⋆⋆ Animations
The character is animated, with several images for each actions.

Level 2 Level 1 must be unlocked to read this section

Click and Run

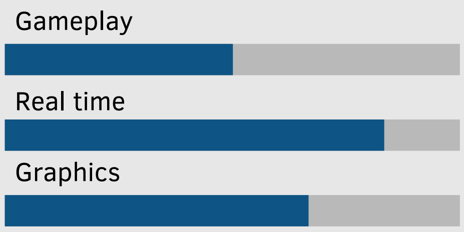
A canabalt-like game

3 people

Project Description

In this game, a player-controlled character is constantly running through a world, always in the same direction. The only control that the player has is to make the character jump to avoid various sorts of threats. The objective is to run for as long as possible. The world is generated procedurally and on the fly.

Skills



(*)[The skill scale is from 0 (Fundamental Awareness) to 6 (Expert).]

Level 1 You may now pursue to the level 1 of the project.

⋆⋆	Procedural world generation Generate an infinite world with a fair amount of variety. The world must consist (at least) of floors and steps and holes.
⋆⋆⋆	World display and start screen The game should be able to display (part of) the world. The start screen should be scrolling from left to right through a procedurally generated world.
⋆	Player A character is running through an empty world, and it can jump and fall; it must not be able to go through walls and floors.
⋆	Complete game mechanics Put the previous two items together and detect death conditions, at least collision but perhaps also falling out of the screen.
⋆	Animation Animate the character, its jumps, and perhaps its death(s).
⋆	Background Generate an infinite background for the world, and display it with a slower scroll, simulating a parallax effect.
⋆	Score Compute a score during the game, increasing (only) when an obstacle is avoided.
⋆	Score UI Display the score during the game, and have a leader board which prompts for a player name when a new high-score is reached.

Level 2 Level 1 must be unlocked to read this section

PAF

le Proof Assistant du Futur

3/4 people

Project Description

A proof assistant helps humans develop formal proofs, through an interaction based on tactics. Famous examples include Coq, Isabelle, Dedukti.

Skills

Languages

Logic

(*)[The skill scale is from 0 (Fundamental Awareness) to 6 (Expert).]

Level 1 You may now pursue to the level 1 of the project.

⋆⋆ Syntax
Implement a datatype and associated operations for the syntax of your logic. You can choose any logic but its syntax should feature at least one binder.

⋆⋆ Interaction loop
Implement a notion of goal and tactic, and an interaction loop which allows to reduce a current goal to new subgoals through tactics.

⋆ Tacticals
Implement common tacticals for composing tactics: sequential composition, choice, repetition, etc.

⋆ Basic tactics
Implement basic tactics covering all rules of your chosen proof system.

⋆ User interface
Wrap everything up with a user interface, which can simply be text-based.

⋆ Pretty-printing
Improve the display of formulas and goals, using indentation, coloring, and possibly optimizing the use of parentheses.

⋆⋆ Advanced tactic
Implement a more complex tactic to provide more automation, e.g. by doing proof search using tacticals.

Level 2 Level 1 must be unlocked to read this section