**Capita Selecta** - Escape Room

Draft

# Overview

The aim is to create a Computer Science (CS) themed Escape Room in which students can apply their theoretical knowledge of CS in an exciting game-like environment. The goal is to deepen the students’ understanding of CS concepts. The room should be *portable* and *low-cost,* so that it can be set up relatively easy by the teacher, regardless of the tools and materials they have to their disposal. The key characteristics of the room are:

* **Target audience**: Middle-school students, aged between 12 and 18.
* **Prior Knowledge**:
  + *Basic Programming* – The students have a basic understanding of programming and are familiar with the concepts of conditional statements, variables, functions and loops.
  + *Logic* – The students have a basic understanding of logic used in CS.
  + *Digital Skills* – The students have the digital skills required to navigate the web, create folders, etc.
* **Team**: Group of 6 to 8 students.

# Planning

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Week** | 45 | 46 | 47 | 48 | 49 | 50 | 51 | 52 |
| 1. Creating draft puzzles, flowchart of EscapeRoom and story |  |  |  |  |  |  |  |  |
| 1. Creating the puzzles |  |  |  |  |  |  |  |  |
| 1. Styling |  |  |  |  |  |  |  |  |
| 1. First test phase |  |  |  |  |  |  |  |  |
| 1. Using feedback for improvement |  |  |  |  |  |  |  |  |
| 1. Second test phase |  |  |  |  |  |  |  |  |
| 1. Writing report on the project |  |  |  |  |  |  |  |  |

1. **Creating draft puzzles, flowchart of the EscapeRoom and story.** In this phase, I would like to create the general setup of the room, i.e. making the (draft) puzzles and fit them into the flow of the room. I want to use a flowchart to describe the way participants are expected to find their way out. Lastly, the story will be thought out to fit the puzzles and the flow of the room.
2. **Creating the puzzles**. In this phase, all the puzzles are elaborated to the point that they can be used in the room. The puzzles will also be tested thoroughly.
3. **Styling**. An EscapeRoom is more than a sequence of puzzles; it’s an experience! Therefore, in this phase I like to work on the styling of the room, i.e. presenting the puzzles in a way that fit the story.
4. **First test phase**. In this phase, the EscapeRoom will be tested for the first time by a class.
5. **Using feedback for improvement**. In this phase, feedback from the first test cycle will be collected and used to improve the room, if necessary.
6. **Second test phase**. In this phase, the EscapeRoom will be tested again. Preferably by someone other than me to ensure the room can also be used by other teachers.
7. **Writing report on the project**. In this phase, I will reflect on the project and create a report, which includes all the EscapeRoom material plus this reflection.

# Backstory

The theme is inspired by the sci-fi novels of Isaac Asimov and the notable Three Laws of Robotics therein.

*The school’s janitor, handyman and overall fix-all, Henry, has recently retired. His office is, however, still filled with junk. The principal orders a small group of students who are in detention to clean out Henry’s office, as a punishment for their bad behaviour. The principal leaves, but tells them that they better do good, otherwise they’ll get even more punishment.*

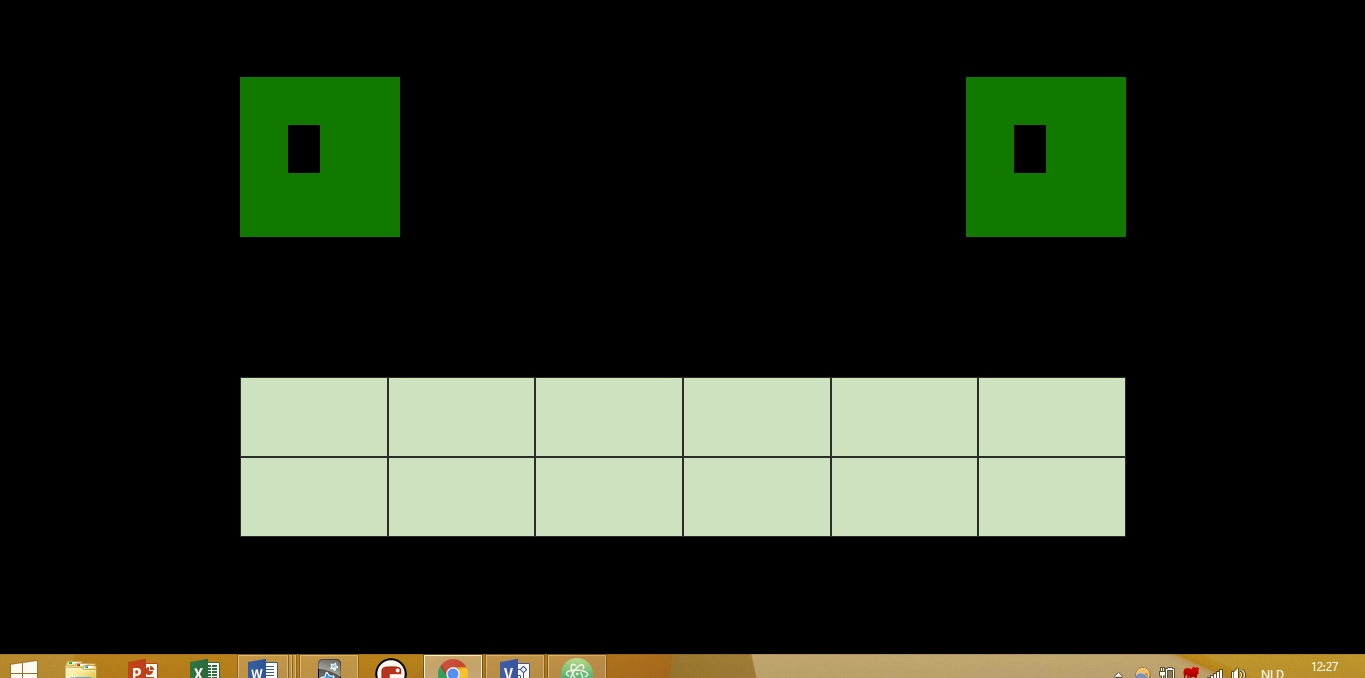
*A great fan of sci-fi, and in particular Isaac Asimov’s “I, Robot”, Henry has been experimenting with building robots himself in his spare time (of which he had plenty!). For the past 4 years he had been working on his masterpiece: The janitor-robot, named “Beezee”. However, when he left the school, he forgot to remove the “beezee.exe” file from his computer. As soon as the students enter the office, the door locks and the students are trapped.*

*According to the first law of robotics (“A robot may not injure a human being or, through inaction, allow a human being to come to harm”), therefore Beezee decides to lock in the students to prevent any harm from befalling them.*

*The students must find a backdoor to manually switch off the robot. However, to do this, they have only 50 minutes. After 50 minutes, Beezee gains complete control and the backdoor will be disabled. But… The problem is that Henry has documented the construction of Beezee horribly. In order to find the backdoor, the students have to go through Henry’s notes and solve puzzles.*

# Puzzle Ideas

The students will be able to interact with the robot Beezee, which will be a simple HTML-page. The robot is made interactive with the use of Javascript/JQuery. The image below shows a first set-up of the robot (as seen on the screen). By clicking on the robot at a certain time/moment/place, the robot itself can be turned into a puzzle.



## Other puzzle ideas:

* Decipher a message encoded with ASCII.
  + To make it more difficult, the students only get an ASCII table with the decimal values, while the encoded message uses hexadecimal values. This requires the students to convert hexadecimal numbers to decimal numbers.
* Decipher an encrypted message, e.g. with the Ceasar-method.
* Use a weight-scale to weigh a certain object. The weight can be a digit code needed to unlock a number lock.
* Play a Morse code for the students to decipher. The Morse code can be played every 2 minutes. The plaintext could be a hint or clue.
* Solving a binary puzzle. The rows and/or columns can be converted to decimal numbers. The sum of either the rows or the columns[[1]](#footnote-1) can be used as a number code for a padlock.
* Conditional puzzle:
* SQL Injection:

# Materials

* Padlock;
* Computer;
* Digiboard;

# Logic Puzzle: Conditionals

If

# LOST Puzzle: Press the right button for 5 minutes

# Morse Code

# ASCII Puzzle

The students find a Decimal-ASCII table somewhere (hidden) in the room. However, the encoded text uses hexadecimals, thus the students need to convert the hexadecimal numbers to decimal numbers in order to find the plaintext.

|  |  |  |
| --- | --- | --- |
|  | English | Dutch |
| Plaintext | Click 28x on the left eye. | *Klik 28x op het linkeroog.* |
| Encoded in HEX |  | 4b 6c 69 6b 20 32 38 78 20 6f 70 20 68 65 74 20 6c 69 6e 6b 65 72 6f 6f 67 2e |

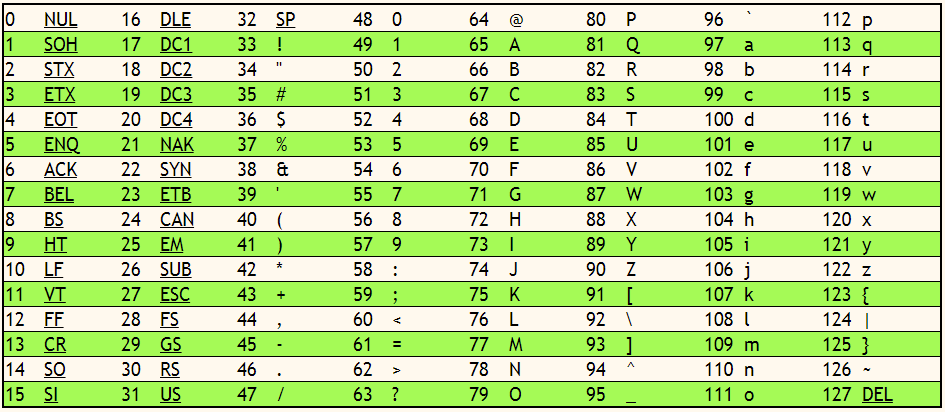


Figure 1 - Decimal to ASCII

Source: <http://www.asciichart.com/>

# Binary Puzzle

The students find a (hidden) binary puzzle in the room. An explanation on how binary puzzles can be solved is also find either with the binary puzzle or somewhere else in the room. By converting the binary rows or columns (the result will be the same) into decimal numbers and adding up the numbers, the students end up with a 3-digit code. This code can be used to open a padlock.

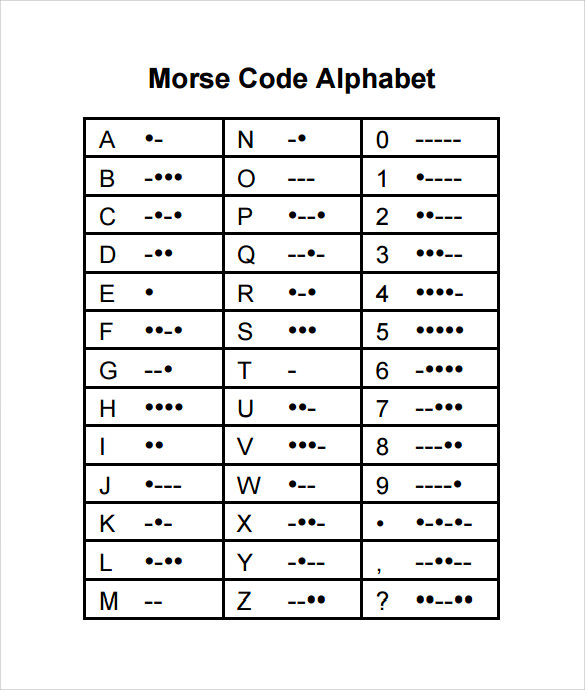
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  | 1 |  |  |
|  |  | 0 |  |  | 1 |
| 0 |  |  |  | 0 |  |
|  | 1 | 1 |  |  |  |
|  |  |  |  |  |  |
| 1 |  |  |  | 0 |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 0 | 1 | 1 | 0 | 22 |
| 1 | 0 | 0 | 1 | 0 | 1 | 37 |
| 0 | 1 | 1 | 0 | 0 | 1 | 25 |
| 0 | 1 | 1 | 0 | 1 | 0 | 26 |
| 1 | 0 | 0 | 1 | 1 | 0 | 38 |
| 1 | 0 | 1 | 0 | 0 | 1 | 41 |
|  |  |  |  |  |  | 189 |

Source: <http://www.binarypuzzle.com/puzzles.php?size=6&level=3&nr=1>

# Morse Code

Every 3 minutes a Morse code is played through the computer. A Morse code alphabet (as seen in the image below) can be found by the students in the room. The students can use the alphabet to decipher the Morse code.



|  |  |  |
| --- | --- | --- |
|  | English | Dutch |
| Plaintext | Click 28x on the left eye. | *Klik 28x op het linkeroog.* |
| Morse Code |  | 4b 6c 69 6b 20 32 38 78 20 6f 70 20 68 65 74 20 6c 69 6e 6b 65 72 6f 6f 67 2e |

# Weighing Scale

# Teeth Code Puzzle

The teeth of the robot can be clicked. Every time a tooth is clicked, the colour of the tooth changes. To solve the puzzle, students have to find the right teeth pattern.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A | B | A | A | C | B |
| B | A | C | C | A | A |

# Binary Search Puzzle

1. The sum of the rows is equal tot he sum of the columns. [↑](#footnote-ref-1)