**Studenten:** Julian Friedrich und Jethro Becker

**Projektname:** Connect 4

**Kurzbeschreibung:** 2 Player Game mit abwechselnder Platzierung der Spielchips

**Programmlibrary:** pygame auf dem RaspberryPI

**Spiel Ablauf:**

In einem 8\* 10 Felder großen Gitter werden durch 2 Spieler abwechselnd verschieden farbige Chips platziert. Nach jedem Durchlauf wird eine Überprüfung vorgenommen ob ein Spieler 4 Chips in einer horizontalen, vertikalen oder diagonalen Reihe aufgereiht hat. Wenn ein Spieler gewonnen hat, erscheint eine entsprechende Meldung auf dem Bildschirm.

**Pygame Library:**

First, we researched different Python libraries and tried out different possibilities. Pygame is a well-documented library and can be programmed intuitively. So, we decided to use Pygame for our Connect 4 Game.

**Implementation (GUI):**

After this we started out with programming the GUI and drawing the game board (Line 9). The init function receives the number of squares for the grid in horizontal and vertical direction. Afterwards the game is initialized with pygame.init (line 11) , the caption of the screen (line 12) and the font to display the winner is set (line 13). The next lines (lines 16- 21) generate the needed variables to draw the grid and the chips, from the number of squares (horizontal, vertical) given when the program is started. Then the background is set. The screen size is set with the given variables and the background is set too white (lines 24 -26). On this background the grid is drawn using two for-loops and the screen is updated with screen.blip (lines 29- 33).

**Gameplay (GUI):**

All the needed draw functions during the game play are defined in the function draw\_circle. (lines 286-….). The variables from line 16-20 are used to define the size of the chips and the middle of the grid squares. The x position as the middle of the grid is defined in line 71. Here a modulo operator is used to switch the x position of mouse (from mouse.getpostition, line 68) to the x position of middle of the square, where the mouse clicked. The y- position is defined in the game logic. See chapter game logic. The color of the circle is defined in the game logic, depending on which players turn it is.

**Gamelogic (GUI):**

In the function “switch player” the player is switched after each turn. This changes the color of the chips from red to blue and back again. This function also changes the caption of the screen in the top left corner.

**Gamelogic (y-position of chips):**

**Gamelogic (Edge Cases):**

The functions: check\_if\_column\_full and check\_if\_board\_full check the expected cases that can occur during gameplay. The first function checks if a column is full. If this is the case it does not allow further Chips to be placed in this column and the user is not switched. The function check\_if\_board\_full chcks if the board is full. If this is the case, the game ends in a tie, the text “No winner” is displayed on the screen and the game closes after 5 seconds.

**Gamelogic (Check for winner):**

While programming the Connect 4 game we defined the flower algorithm to check for 4 chips in a row in horizontal, vertical and in diagonal direction. The flower algorithm is displayed in the following picture:

The flower algorithm is defined in the functions: check\_horizontal, check\_vertical, check\_diagonal and check\_inverted\_diagonal. Each of these functions checks for four chips in a row for the given direction.

**Further Possibilities:**

Given more time following list of features could be implemented.

- GUI: User prompt at begin of program with three different board sizes.

- GUI: User prompt at the end of game with the two possibilities end or restart. If end: end of program. If restart: restart the game.

- Ask for player names and set these at the beginning of the game

- Pick colors for chips from a list and convert these to RGB values to set chip color.