**Programming Tools in Artificial Intelligence**

**Course Code: (CT5132/CT5148)**

**Assignment 3**

Students Name:

1. Moiz Meyaji (Student ID: 19233048)
2. Waseem Shariff (Student ID: 19233839)

Tasks:

**Task 1**: 6f8cd79b.json

**Task 2**: 25d8a9c8.json

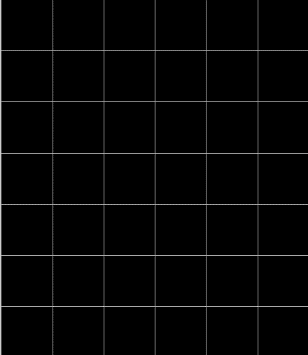
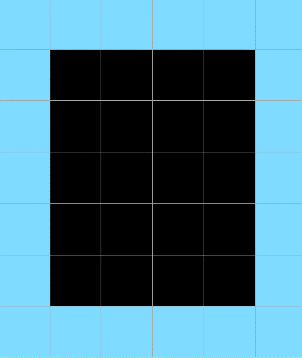
**Task 3**: 794b24be.json

Link to the GitHub: <https://github.com/MoizSM/ARC>

--------------------------------------------------------------------------------------

**Task 1: 6f8cd79b.json**

**Input**: **Output:**



**Pattern Logic:**

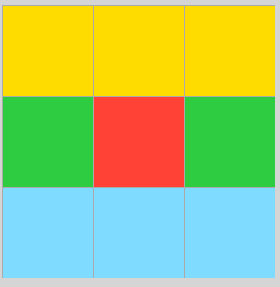
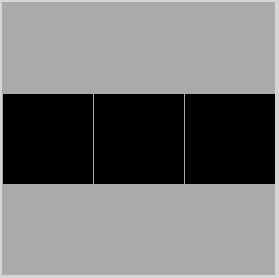
From the above representation of the images we can observe that, the input image is a blank grid and the expected output is the grid with blue color at its first and last row and column. We have to find a logic such that, any given input with blank grid, outputs a grid with a color at its boundaries.

**Code Logic:**

We have defined a function **solve,** which consists of the logic of this problem. In the solve function we first take the input of all the training cases and all the test cases. We then solve each case by displaying their specific outputs. Using for loop we first took the rows of the grid, then we colored the first and last row of the grid. Now for the first and last column of the grid, we use nested for loop, where we get the elements of each row separately. Then we assign the colors to the first and last element of each row. The function solve() produces the correct output for all the test and training cases

**Task 2: 25d8a9c8.json**

**Input: Output:**



**Pattern Logic:**

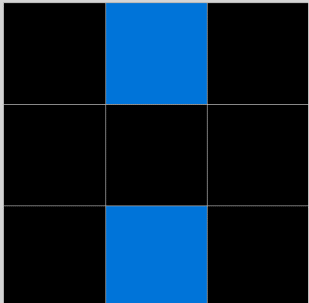
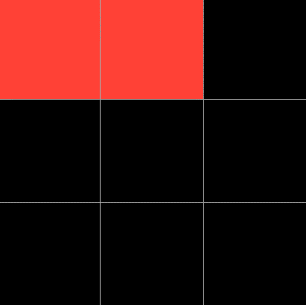
As we can see from the above representations that the input grid contains a random collection of colors. However, the grid contains certain number of rows (1st and 2nd in this case) which consist of the same color. For any grid having similar patterns the output produced is a grid where the rows having same colors are converted to grey while the remaining elements of the grid are converted to black.

**Code Logic:**

In the code for this task, the solve function parses the json file and runs a for-loop on all the training and testing input grids. The for-loop will check if the elements of every row matches the first element of that row. This is done using boolean values (**check**). If all the values of the row are same then **check = True** and the elements of the row are colored grey, else the **check = False,** and the elements of the row are colored black. The function solve() produces the correct output for all the test and training cases.

**Task 3: 794b24be.json**

**Input: Output:**



**Pattern Logic:**

From the above representation of the images we can observe that, the input grid consists of ‘n’ number of scattered blue (8) colored elements in a grid. The output grid produces red (2) colored elements in a sequence of line. However, the if n < = 3, then output elements will be within the first row, else if n >3, then the 4th element will be in the second row and second column.

**Code Logic:**

The solve function parses the json file and runs a for-loop on all the training and testing inputs. We are storing the total number of blue elements of the grid in **count.** We are then using an **if** condition to color the elements of the first row of the output grid red (2) for the count value.

**Summary**

The tasks given in the ARC Corpus by Chollet have provided us with the opportunity to implement critical thinking and logical reasoning skills. It was ecstatic to create our own algorithms to solve different tasks from the corpus.

The libraries we used for our solutions were:

* **sys:** It’s a module in the standard library that we imported to utilize the json file that is passed as a parameter in the command line while executing the python program.
* **json:** We imported the json module to parse the json file that is read by the python program
* **NumPy:** We imported the NumPy library to convert the json grids into a NumPy array. This helped us utilize the NumPy features to work with he grids to solve the task.

Python helps to implement easily readable code and supports large number of libraries. All our solutions were implemented using the libraries mentioned above. The tasks above provided different challenges that were based on patterns of colors in a grid. Each task had different outputs based on the challenge. Our algorithms for every task mainly consisted of **for-loops** to create solutions for the challenge.

**Contributions**

1. **Moiz Meyaji –** I worked on creating the solution for task 2 (25d8a9c8.json). I also collaborated to solve task 3 (794b24be.json)with Waseem. We created and implemented the logic of the solution. I wrote the requirements of my tasks and the code logic of task 3 along with the summary.
2. **Waseem Shariff-** I worked on creating the solution for task 1 (6f8cd79b.json). I also collaborated to solve task 3 (794b24be.json) with Moiz. We created and implemented the logic for the solution. I wrote the requirements for my task and the pattern logic for task 3 in the report.