HOW SHOULD I SAVE THE PRINCESS

Here's the Python function display Path to Princess that calculates and prints the moves required to rescue Princess Peach. Each move is printed on a new line, and the function is well-commented to explain each step.

➢ HERE IS THE PYTHON CODE FOR THIS CONDITION: -

```
def display_Path_to_Princess(N, grid):

    """

    Function to determine and print the shortest path to rescue the princess.

        Args:

        N (int): The size of the grid (N x N). N is always odd.

        grid (list): A 2D list representing the grid with 'm' for Mario and 'p' for Princess.

    Prints:

        Moves (LEFT, RIGHT, UP, DOWN) on separate lines to reach the princess.

    """

    # Locate the center of the grid (where Mario starts)

    center = (N // 2, N // 2)  # Mario's position



    # Locate the princess ('p') in the grid

    for i in range(N):

        for j in range(N):

            if grid[i][j] == 'p':

                princess = (i, j)

                break
    # Calculate the moves required to reach the princess
    moves = []
    row_diff = princess[0] - center[0]  # Vertical distance
    col_diff = princess[1] - center[1]  # Horizontal distance

    # Add UP or DOWN moves based on row_diff
    if row_diff < 0:  # Princess is above Mario
        moves.extend(["UP"] * abs(row_diff))
    elif row_diff > 0:  # Princess is below Mario
        moves.extend(["DOWN"] * abs(row_diff))

    # Add LEFT or RIGHT moves based on col_diff
    if col_diff < 0:  # Princess is to the left of Mario
        moves.extend(["LEFT"] * abs(col_diff))
    elif col_diff > 0:  # Princess is to the right of Mario
        moves.extend(["RIGHT"] * abs(col_diff))

    # Print all moves on separate lines
    for move in moves:
        print(move)
```

# SAMPLE INPUT : -

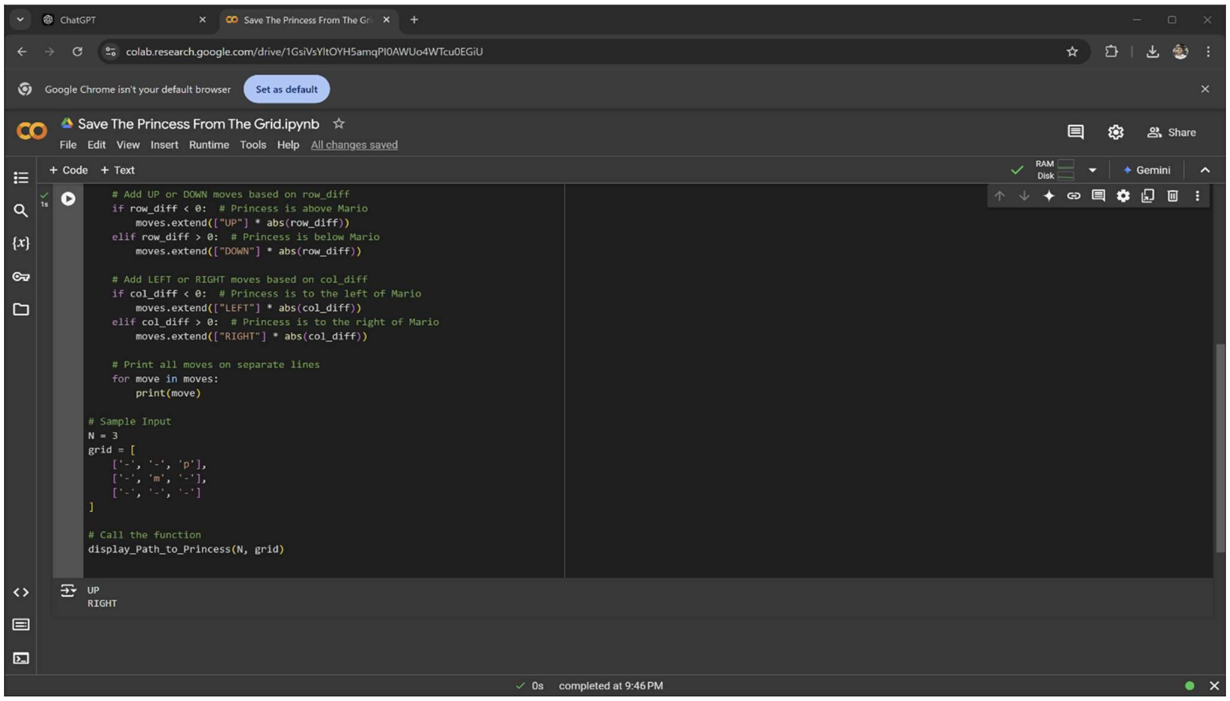
```
N = 3
grid = [
    ['-', '-', 'p'],
    ['-', 'm', '-'],

    ['-', '-', '-']

]
```

# CALL THE FUNCTION : -

display_Path_to_Princess(N, grid)



ChatGPT
Save The Princess From The Gr
colab.research.google.com/drive/1GsiVsYltOYH5amqPI0AWUo4WTcu0EGiU
Google Chrome isn't your default browser
Set as default
Save The Princess From The Grid.ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
Share
Gemini

```
    # Add UP or DOWN moves based on row_diff
    if row_diff < 0:  # Princess is above Mario
        moves.extend(["UP"] * abs(row_diff))
    elif row_diff > 0:  # Princess is below Mario
        moves.extend(["DOWN"] * abs(row_diff))

    # Add LEFT or RIGHT moves based on col_diff
    if col_diff < 0:  # Princess is to the left of Mario
        moves.extend(["LEFT"] * abs(col_diff))
    elif col_diff > 0:  # Princess is to the right of Mario
        moves.extend(["RIGHT"] * abs(col_diff))

    # Print all moves on separate lines
    for move in moves:
        print(move)

# Sample Input
N = 3
grid = [
    ['-', '-', 'p'],
    ['-', 'm', '-'],
    ['-', '-', '-']
]

# Call the function
display_Path_to_Princess(N, grid)
```

```
UP
RIGHT
```

- **Center Position**: The starting position of Mario is calculated as (N//2, N//2) since N is always odd.

- **Princess Location**: The grid is scanned to find the position of the princess ('p').

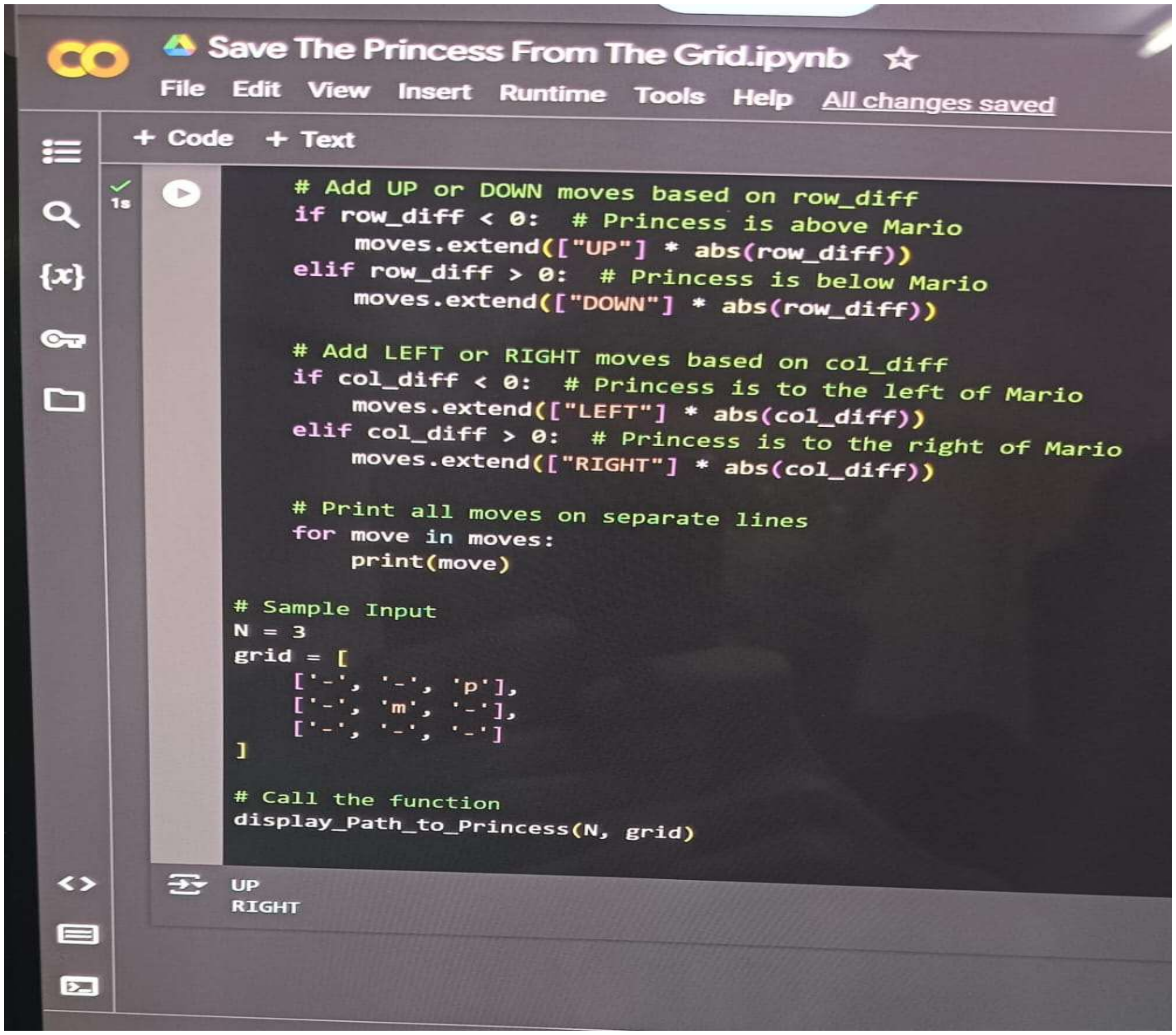
- **Row and Column Difference**:

  row_diff: Determines if Mario needs to move up or down.

  col_diff: Determines if Mario needs to move left or right.

- **Move Calculation**:
  Use the sign of `row_diff` and `col_diff` to determine the directions (`UP`, `DOWN`, `LEFT`, `RIGHT`) and the number of moves needed.

- **Output**: Each move is printed on a new line for the shortest path


Save The Princess From The Grid.ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
1s

```
        # Add UP or DOWN moves based on row_diff
        if row_diff < 0:  # Princess is above Mario
            moves.extend(["UP"] * abs(row_diff))
        elif row_diff > 0:  # Princess is below Mario
            moves.extend(["DOWN"] * abs(row_diff))

        # Add LEFT or RIGHT moves based on col_diff
        if col_diff < 0:  # Princess is to the left of Mario
            moves.extend(["LEFT"] * abs(col_diff))
        elif col_diff > 0:  # Princess is to the right of Mario
            moves.extend(["RIGHT"] * abs(col_diff))

        # Print all moves on separate lines
        for move in moves:
            print(move)

# Sample Input
N = 3
grid = [
        ['-', '-', 'p'],
        ['-', 'm', '-'],
        ['-', '-', '-']
]

# Call the function
display_Path_to_Princess(N, grid)
```

```
UP
RIGHT
```