|  |
| --- |
| # SNAKES GAME |
|  |

|  |
| --- |
| # Use ARROW KEYS to play, SPACE BAR for pausing/resuming and Esc Key for exiting |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| import curses |
|  |

|  |
| --- |
| from curses import KEY\_RIGHT, KEY\_LEFT, KEY\_UP, KEY\_DOWN |
|  |

|  |
| --- |
| from random import randint |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| curses.initscr() |
|  |

|  |
| --- |
| win = curses.newwin(20, 60, 0, 0) |
|  |

|  |
| --- |
| win.keypad(1) |
|  |

|  |
| --- |
| curses.noecho() |
|  |

|  |
| --- |
| curses.curs\_set(0) |
|  |

|  |
| --- |
| win.border(0) |
|  |

|  |
| --- |
| win.nodelay(1) |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| key = KEY\_RIGHT # Initializing values |
|  |

|  |
| --- |
| score = 0 |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| snake = [[4,10], [4,9], [4,8]] # Initial snake co-ordinates |
|  |

|  |
| --- |
| food = [10,20] # First food co-ordinates |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| win.addch(food[0], food[1], '\*') # Prints the food |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| while key != 27: # While Esc key is not pressed |
|  |

|  |
| --- |
| win.border(0) |
|  |

|  |
| --- |
| win.addstr(0, 2, 'Score : ' + str(score) + ' ') # Printing 'Score' and |
|  |

|  |
| --- |
| win.addstr(0, 27, ' SNAKE ') # 'SNAKE' strings |
|  |

|  |
| --- |
| win.timeout(150 - (len(snake)/5 + len(snake)/10)%120) # Increases the speed of Snake as its length increases |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| prevKey = key # Previous key pressed |
|  |

|  |
| --- |
| event = win.getch() |
|  |

|  |
| --- |
| key = key if event == -1 else event |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| if key == ord(' '): # If SPACE BAR is pressed, wait for another |
|  |

|  |
| --- |
| key = -1 # one (Pause/Resume) |
|  |

|  |
| --- |
| while key != ord(' '): |
|  |

|  |
| --- |
| key = win.getch() |
|  |

|  |
| --- |
| key = prevKey |
|  |

|  |
| --- |
| continue |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| if key not in [KEY\_LEFT, KEY\_RIGHT, KEY\_UP, KEY\_DOWN, 27]: # If an invalid key is pressed |
|  |

|  |
| --- |
| key = prevKey |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| # Calculates the new coordinates of the head of the snake. NOTE: len(snake) increases. |
|  |

|  |
| --- |
| # This is taken care of later at [1]. |
|  |

|  |
| --- |
| snake.insert(0, [snake[0][0] + (key == KEY\_DOWN and 1) + (key == KEY\_UP and -1), snake[0][1] + (key == KEY\_LEFT and -1) + (key == KEY\_RIGHT and 1)]) |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| # If snake crosses the boundaries, make it enter from the other side |
|  |

|  |
| --- |
| if snake[0][0] == 0: snake[0][0] = 18 |
|  |

|  |
| --- |
| if snake[0][1] == 0: snake[0][1] = 58 |
|  |

|  |
| --- |
| if snake[0][0] == 19: snake[0][0] = 1 |
|  |

|  |
| --- |
| if snake[0][1] == 59: snake[0][1] = 1 |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| # Exit if snake crosses the boundaries (Uncomment to enable) |
|  |

|  |
| --- |
| #if snake[0][0] == 0 or snake[0][0] == 19 or snake[0][1] == 0 or snake[0][1] == 59: break |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| # If snake runs over itself |
|  |

|  |
| --- |
| if snake[0] in snake[1:]: break |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| if snake[0] == food: # When snake eats the food |
|  |

|  |
| --- |
| food = [] |
|  |

|  |
| --- |
| score += 1 |
|  |

|  |
| --- |
| while food == []: |
|  |

|  |
| --- |
| food = [randint(1, 18), randint(1, 58)] # Calculating next food's coordinates |
|  |

|  |
| --- |
| if food in snake: food = [] |
|  |

|  |
| --- |
| win.addch(food[0], food[1], '\*') |
|  |

|  |
| --- |
| else: |
|  |

|  |
| --- |
| last = snake.pop() # [1] If it does not eat the food, length decreases |
|  |

|  |
| --- |
| win.addch(last[0], last[1], ' ') |
|  |

|  |
| --- |
| win.addch(snake[0][0], snake[0][1], '#') |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| curses.endwin() |
|  |

|  |
| --- |
| print("\nScore - " + str(score)) |
|  |

print("http://bitemelater.in\n")