PROJECT 1:

NUMBER GUESSING GAME

> Build a Number guessing game, in which the user selects a range.
> Let's say User selected a range, i.e., from A to B, where A and B belong to Integer.
> Some random integer will be selected by the system and the user has to guess that integer in the minimum number of
guesses

Explaination:
If the User inputs range, let's say from 1 to 100. And compiler randomly selected 42 as the integer. And now the guessing
game started, so the user entered 50 as his/her first guess. The compiler shows "Try Again! You guessed too high". That's
mean the random number (i.e., 42) doesn't fall in the range from 50 to 100. That's the importance of guessing half of the
range. And again, the user guesses half of 50 (Could you tell me why?). So the half of 50 is 25. The user enters 25 as his/her
second guess. This time compiler will show, "Try Again! You guessed too small". That's mean the integers less than 25
(from 1 to 25) are useless to be guessed. Now the range for user guessing is shorter, i.e., from 25 to 50. Intelligently!
The user guessed half of this range, so that, user guessed 37 as his/her third guess. This time again the compiler shows
the output, "Try Again! You guessed too small". For the user, the guessing range is getting smaller by each guess. Now, the
guessing range for user is from 37 to 50, for which the user guessed 43 as his/her fourth guess. This time the compiler will
show an output "Try Again! You guessed too high". So, the new guessing range for users will be from 37 to 43, again for which
the user guessed the half of this range, that is, 40 as his/her fifth guess. This time the compiler shows the output, "Try Again!
You guessed too small". Leaving the guess even smaller such that from 41 to 43. And now the user guessed 41 as his/her
sixth guess. Which is wrong and shows output "Try Again! You guessed too small". And finally, the User Guessed the right
number which is 42 as his/her seventh guess.

Total Number of Guesses = 7

PROJECT 2:

ROCK PAPER SCISSORS

This program or a mini-game is designed when you don't have anyone to play or you are under lockdown alone. There are a
number of functions that this program requires so let us have an overview of each.

> a random function: to generate rock, paper, or scissors.
> valid function: to check the validity of the move.
> result function: to declare the winner of the round.
> scorekeeper: to keep track of the score.

The program requires the user to make the first move before it makes one the move. Once the

move is validated the input is
evaluated, the input entered could be a string or an alphabet. After evaluating the input string a winner is decided by the
result function and the score of the round is updated by the scorekeeper function.

What Is Rock Paper Scissors?
You may have played rock paper scissors before. Maybe you've used it to decide who pays for dinner or who gets first choice
of players for a team.

If you're unfamiliar, rock paper scissors is a hand game for two or more players. Participants say "rock, paper, scissors" and
then simultaneously form their hands into the shape of a rock (a fist), a piece of paper (palm facing downward), or a pair of
scissors (two fingers extended). The rules are straightforward:

Rock smashes scissors.
Paper covers rock.
Scissors cut paper.

PROJECT 3:

TIC TAC TOE

It is a two-player game and consists of a nine-square grid. Each player chooses their move and with O or X and marks their
square one at each chance. The player who succeeds in making their marks all in one line whether diagonally, horizontally, or
vertically wins. The challenge for the other player is to block the game for their opponent and also to make their chain.

The board is numbered like the keyboard's number pad. And thus, a player can make their move in the game board by entering
the number from the keyboard number pad.