

2025-01-17 20:35

Status: [#programming](#) [#problem-solving](#) [#project-building](#) [#Golang](#)

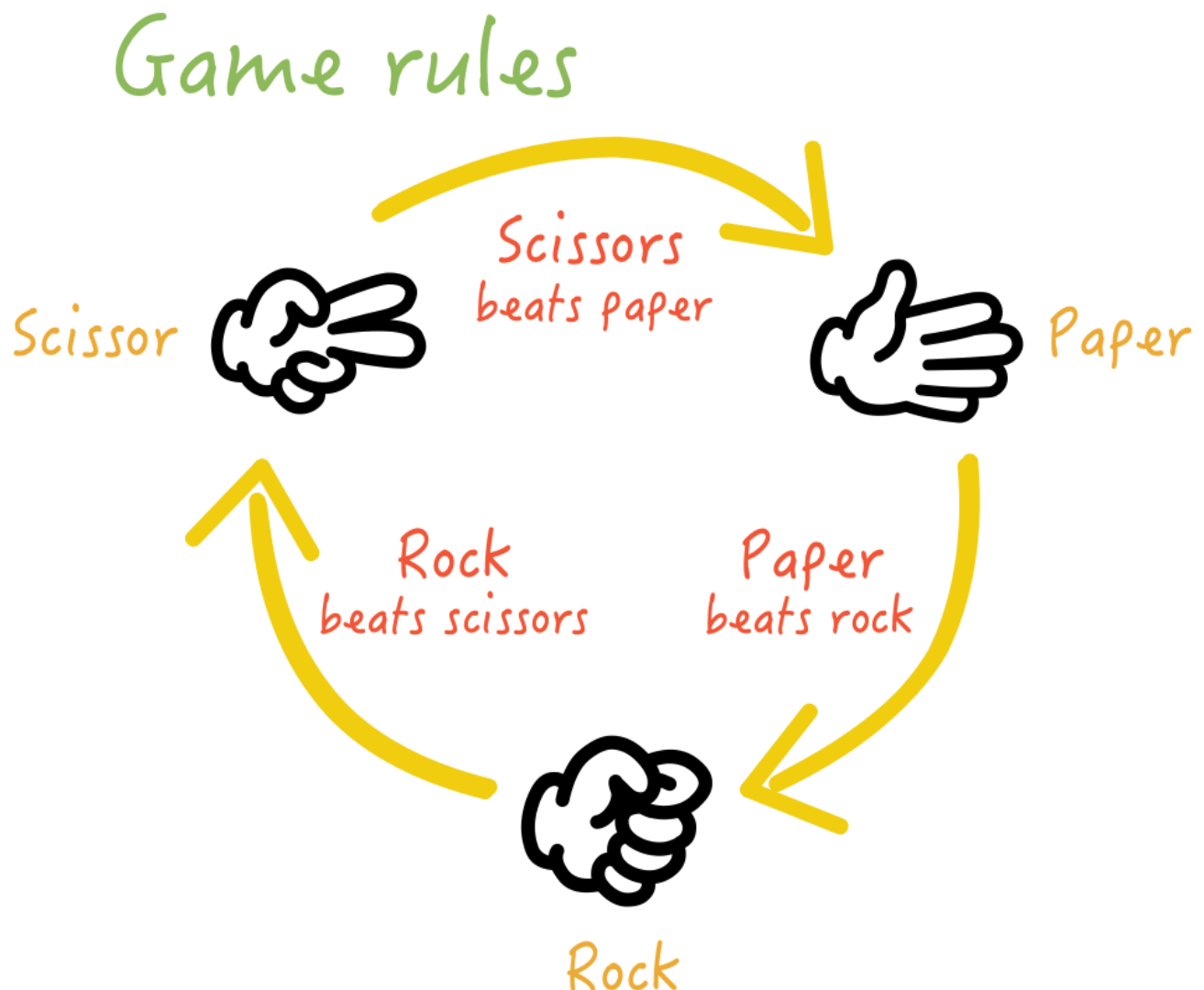
Tags:

All small Golang projects Details

1. Building the "Rock, Paper, Scissors, Lizard, Spock" game

Part One

We all know about the Simpler one. Most of us played the version "Rock, Paper and Scissors" at any point of our life. So, here's a picture of how "Rock, Paper, Scissors" work.



You can already see the explanation of the picture above.

So, here's our first part of the project:

Input:

The program you will build will ask you for an input which would your move in the game. So only give "**Rock**" or "**Paper**" or "**Scissor**" as your game input. If you input any invalid move then the program will give you the output : "**Invalid move. Game over**".

So that's it for the input.

Now, to make the game fair your program have to randomly generate the move for the Computer(It make sense cause when two player are playing this game, no one knows what the other player will play, but as the computer is taking your move as input it already knows that what it should move to win the game). So, that's why we will make a random functionality to the program which will already generate the move for the Computer and store it in a secret variable which you won't able to say before giving your input. Then, your program should print that variable (The move of the Computer) and now that you have both input. You can determine who the winner is!

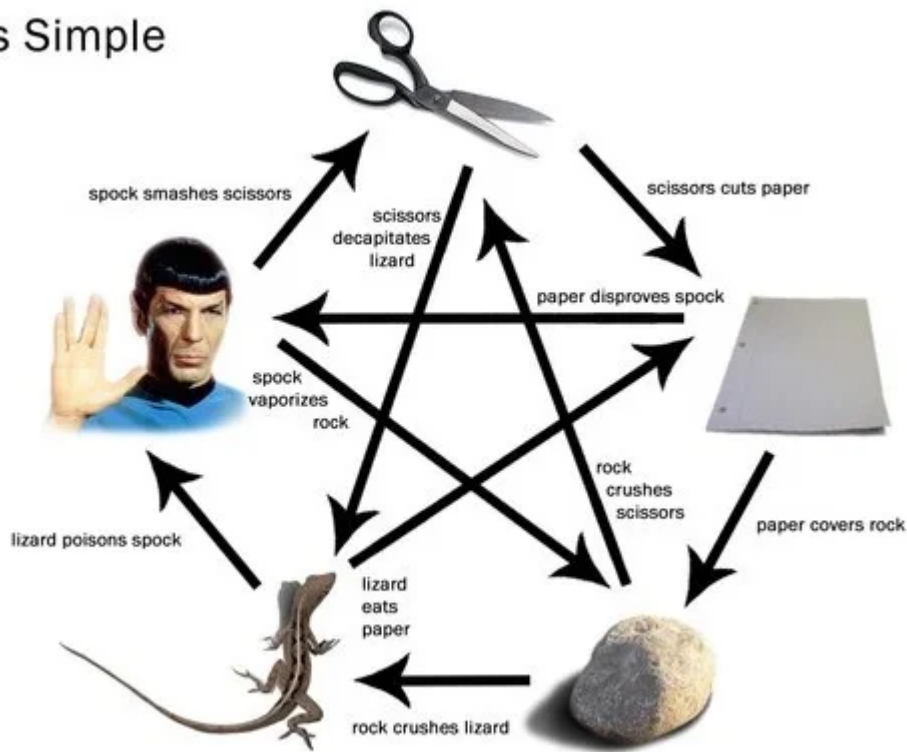
Output:

So as for the output you can print "You won! Congratulations!" if the **user** (user of the program which can be you as well) **win**. And maybe print "You lost! hahaha, you suck!" if our user **lose**. But there is another outcome, right? Which is **Draw**. When the user's move and computer's move matches the print "Damn, it's a draw. Let's play again!" and then your program will loop back and ask the user for the Input (your move) again. This Loop part you can ignore if you don't yet wanna work with Loops

Part 2

Now, this part is real fun. All you have to do is modify your program you built before a bit and Make it to the "Rock, Paper, Scissors, Lizard, Spock" from the simpler "Rock, Paper, Scissors"

Its Simple



Let's give you guys a clear picture!

Watch this video for reference: https://www.youtube.com/watch?v=_PUEoDYpUyQ&ab_channel=Qashi

Don't worry! I can assure you it's a total safe link. It's cut scene from a famous show "The big bang theory". You can safe check the link if you want to and don't trust me (you shouldn't)

Anyway, same **Input** and **Output** method as before and it's a plus if you can loop back the program when it's a draw.

A better explanation

Rock lose against 2:

1. Paper
2. Spock

Rock win against 2:

1. Scissor
2. Lizard

Scissor lose against 2:

1. Rock
2. Spock

Scissor win against 2:

1. Paper
2. Lizard

Paper lose against 2:

1. Lizard
2. Scissor

Paper win against 2:

1. Rock
2. Spock

Lizard lose against 2:

1. Rock
2. Scissor

Lizard win against 2:

1. Spock
2. Paper

Spock lose against 2:

1. Paper
2. Lizard

Spock win against 2:

1. Scissor
2. Rock

Tips:

For now you can just make a CLI version of this game project. But once you have the skill and tools you can make an UI as well and make it a bit bigger project in the future.

My Project link (source code): <https://pastebin.com/U5UXcmtn>

2. Number Guessing Game with Golang

So, It's really simple. Your program have to generate a random number between 1-100. And the User have to guess the number correctly to win the game. This is the idea at least. Each Time the user will guess the number your program will give them hint like "Your guess is too high" or "Your guess is too low" etc. When you will guess the exact number you will win the game.

But Let's make it more interesting. For example your program will give the user the opportunity to choose the difficulty of this game. To do that you might wanna increase the range of the number (e.g make 1-500) and also maybe give the user a limited amount of turn.

So for my project my program will give the user to choose from some difficulty levels:

Difficulty Name	Range of number to randomly generate from	How Many turns the user will be given
Casual Level	1 - 50	8
Standard Level	1 - 100	6
Challenging Level 1	1 - 100	5
Challenging Level 2	1 - 100	4
Extreme Level	1 - 500	8
Impossible Level	1 - 1000	8

Also for each level you can make different scoring system as well which you can implement the way you want. For example, In the Casual level the user will be given 8 chances to guess the secret number. But if you can guess it in

the 5th turn that means you saved 3 turns. and for each saved turn you get $3 \times 10 = 30$ points bonus!! So, Let's make another example table for the scoring system.

Difficulty Name	The points you will get if you win	Bonus point for each turn saved
Casual Level	50	10
Standard Level	100	50
Challenging Level 1	300	100
Challenging Level 2	500	200
Extreme Level	1000	500
Impossible Level	3000	750

Now, your project is much better and cool!

When you will learn Backend Development and will learn to manage a database you can even make an web app for this game with an UI and there everyone will able to play this game. All the info of the player will be collected in the database. You can even make top 10 high score list for a specific user and also create leader-board to show who has the best score in that web app as well as creating a ranking system. You can do so much with programming!!!

So, yeah go on! Now try to add more feature. Make your project look like an art!

My Project Link (Source Code): <https://pastebin.com/e59qCHkR>



3. Building a Simple Calculator (CLI)

So the idea is to create a command-line calculator that takes user input for numbers and an operator (+, -, *, /) to perform calculations.

But you can add so much more here! It's Totally up to you!

This is one of the most easiest project everyone can do.

For my project I wanna add some more functionality to my Calculator. For example, My Calculator will be able to calculate the root value ($\sqrt{\quad}$) of

something. Also, will be able to square a given number and take any number  as power of that given number where my program will take  as input as well.

There are so many different versions a CLI calculator can be made. For my project, I will try to make CLI text user-friendly and make every action easy and obvious to perform. The user will be given their name, then will be asked to give integer/floating number as input and after taking each number as input it will ask which operation it should perform. There will be two modes. One is basic where one operation can be done at a time. Another one will be complex where multiple operations and multiple inputs can be given at the same time.

So, let's stop yapping now and get to it! The project will speak for itself so I don't really have to explain what it will do.

My project Link(Source code): <https://pastebin.com/h0CGmHDw>

References

1. I used ASCII art from here: [Patorjk](#)
2. Any other future references will go here.
3. Shunting Yard Algorithm [Youtube Video](#)