# Technical test

Since no one remembers the card ranking you will have to write a program that compares poker hands and determines a winner.

## Requirements

A hand is any Type that implements `ComparableHand` and can be initialized with a string of 5 cards, for example:

``` go

hand := NewHand("KS 2H 5C JD TD")

```

To solve this problem you must design a type that implements the `ComparableHand` interface, and in particular the `CompareWith(compareTo ComparableHand) (result int)` method. This method should be responsible for determining which of two hands is the winner, indicated by `result`:

\* `0` for a TIE

\* `1` for a WIN

\* `2` for a LOSS

You must also complete the implementation of `NewHand(hand string) ComparableHand{}` which accepts a string that defines a single poker hand, as defined below:

\* A space is used as card separator

\* Each card consists of two characters

\* The first character is the value of the card, valid characters are: `2`, `3`, `4`, `5`, `6`, `7`, `8`, `9`, `T`(en), `J`(ack), `Q`(ueen), `K`(ing), `A`(ce)

\* The second character represents the suit, valid characters are: `S`(pades), `H`(earts), `D`(iamonds), `C`(lubs)

\_Note: You may need to use a type assertion inside the `CompareWith` method in order to access any fields of your custom type, but this is not essential for a good solution.\_

You are free to architect your code any way you want: adding structs, interfaces or constants as you please. However, you \*\*must\*\* adhere to the two requirements above. You may use any other libraries that you feel are relevant to solve this problem.

The ranking of the hands should follow the [Texas Hold'em rules](https://www.partypoker.com/how-to-play/hand-rankings.html)

## Tests

Some unit tests have been included in the code skeleton. You can run them as a script: `go test`. Writing more tests is welcome :)

Good luck ;)