1 String-Addition

1.1 Aufgabe

Implementiere eine Funktion `greetPlayerAskChoiceHelper`, die einen Namen als String übergeben bekommt und einen String zurückgibt, in dem die Person gebeten wird, sich für Schere, Stein oder Papier zu entscheiden.

```
greetPlayerAskChoiceHelper(”Anna”)
```

```
Hey Anna! Please choose rock, paper or scissors!
```

```
greetPlayerAskChoiceHelper(”Timo”)
```

```
Hey Timo! Please choose rock, paper or scissors!
```

2 String-Addition und Typkonversion

2.1 Aufgabe

Implementiere eine Funktion `showRoundNumber`, die zwei Integer als Argumente übergeben bekommt. Dabei handelt es sich um die Runde die gerade gespielt wird und die Anzahl der Runden, die insgesamt gespielt werden. Die Funktion gibt einen String zurück, in dem beide Informationen enthalten sind.

```
showRoundNumber(1, 3)
```

```
1. of 3 Rounds
```

```
showRoundNumber(4, 5)
```

```
4. of 5 Rounds
```

2.2 Aufgabe

Implementiere eine Funktion `greetPlayerAskNameHelper`. Dieser Funktion wird die Nummer eines Spielers als Integer übergeben. Sie gibt einen String zurück, in dem der Spieler begrüßt und nach seinem Namen gefragt wird.

```
greetPlayerAskNameHelper(1)
```

```
Hello Player 1! What's your name?
```

```
greetPlayerAskNameHelper(2)
```

```
Hello Player 2! What's your name?
```

3 Bedingungen

3.1 Aufgabe

Implementiere eine Funktion `showPlayerPoints`. Dieser Funktion wird ein Name als String und eine Punktezahl als Integer übergeben. Sie gibt einen String zurück, in dem steht wie viele Punkte die Person hat.

```
showPlayerPoints("Kevin", 1)
```

```
Kevin has 1 point
```

```
showPlayerPoints("Barbara", 3)
```

```
Barbara has 3 points
```

Beachte, dass im ersten Beispiel `point` und nicht `points` zu sehen ist.

3.2 Aufgabe

Implementiere eine Funktion `showPlayersPoints`. Dieser Funktion werden die Namen der Spieler als Strings und ihre Punktezahlen als Integer übergeben. Sie gibt einen String zurück, in dem steht wie viele Punkte die ide Spieler haben. Ziwschen beiden Information steht das Zeichen `\n`.

```
showPlayersPoints("Kevin", "Barbara", 1,  3)
```

```
Kevin has 1 point
Barbara has 3 points
```

Nutze `showPlayerPoints`!

3.3 Aufgabe

Implementiere eine Funktion `computeNewPoints`. Dieser wird der Gewinner der letzten Runde, die Nummer eines Spielers und die Punktzahl des Spielers mit dieser Nummer übergeben. Die einzigen Möglichkeiten für das erste Argument sind `"1"`, `"2"` und `"Nobody"`. Die Funktion gibt die neue Punktzahl des Spielers zurück.

```
computeNewPoints("1", 1, 3)
```

```
4
```

```
computeNewPoints("2", 1, 3)
```

```
3
```

```
computeNewPoints("Nobody", 1, 1)
```

```
1
```

3.4 Aufgabe

Implementiere eine Funktion `computeRoundWinner`. Dieser werden die Eingaben der beiden Spieler als Strings übergeben. Sie gibt zurück, wer die Runde gewonnen hat. Die einzigen möglichen Argumente sind `"rock"`, `"paper"` und `"scissors"`. Die einzigen möglichen Rückgabewerte sind `"1"`, `"2"` und `"Nobody"`.

```
computeRoundWinner("rock", "scissors")
```

```
1
```

```
computeRoundWinner("rock", "paper")
```

```
2
```

```
computeRoundWinner("rock","rock")
```

```
Nobody
```

3.5 Aufgabe

Implementiere eine Funktion `showWinnerWithNameHelper`. Dieser wird übergeben, wer das Spiel/die Runde gewonnen hat. Die einzigen Möglichkeiten sind `"1"`, `"2"` und `"Nobody"`. Außerdem werden die Namen der Spieler und ein Boolean übergeben. Wenn dieses `true` ist soll als String zurückgegeben werden welcher Spieler, das Spiel gewonnen hat. Ansonsten wird zurückgegeben welcher Spieler die Runde gewonnen hat. In dem String werden die übergebenen Namen verwendet.

```
1 showWinnerWithNameHelper("1", "Grace", "Alan", true)
```

```
Grace wins this game!
```

```
1 showWinnerWithNameHelper("2", "Grace", "Alan", false)
```

```
Alan wins this round!
```

```
1 showWinnerWithNameHelper("Nobody", "Grace", "Alan", false)
```

```
Nobody wins this round!
```

3.6 Aufgabe

Implementiere eine Funktion `showRoundWinnerWithName`. Dieser wird übergeben wer das Spiel gewonnen hat. Die einzigen Möglichkeiten sind `"1"`, `"2"` und `"Nobody"`. Außerdem werden die Namen der Spieler übergeben. Die Funktion gibt einen String zurück in dem steht, wer die Runde gewonnen hat. In dem String wird der Name des Spielers verwendet.

```
1 showRoundWinnerWithName("1", "Grace", "Alan")
```

```
Grace wins this round!
```

```
1 showRoundWinnerWithName("2", "Grace", "Alan")
```

```
Alan wins this round!
```

```
1 showRoundWinnerWithName("Nobody", "Grace", "Alan")
```

```
Nobody wins this round!
```

Nutze `showWinnerWithNameHelper`!

3.7 Aufgabe

eine Funktion `showGameWinnerWithName`. Dieser werden Namen der Spieler als Strings und ihre die Punktezahlen als Integer übergeben. Die Funktion gibt einen String zurück in dem steht, wer das Spiel gewonnen hat.

```
showGameWinnerWithName(”Grace”, ”Alan”, 1, 2)
```

```
Alan wins this game!
```

```
showGameWinnerWithName(”Grace”, ”Alan”, 5, 0)
```

```
Grace wins this game!
```

```
showGameWinnerWithName(”Grace”, ”Alan”, 1 , 1)
```

```
Nobody wins this game!
```

> Nutze `showWinnerWithNameHelper`!

4 Eingabe

In diesem Abschnitt wird mit den bereits definierten Funktionen das vollständige Spiel implementiert. Ein Durchlauf des Spiels kann folgendermaßen aussehen:

```
Hello Player 1! What's your name?
Grace
Hello Player 2! What's your name?
Alan
How many rounds do you want to play?
2
1. of 2 Rounds
Hey Grace! Please choose rock, paper or scissors!
rock
Hey Alan! Please choose rock, paper or scissors!
paper
Alan wins this round!
Grace has 0 points
Alan has 1 point
2. of 2 Rounds
Hey Grace! Please choose rock, paper or scissors!
paper
Hey Alan! Please choose rock, paper or scissors!
```

```
19 paper
20 Nobody wins this round!
21 Grace has 0 points
22 Alan has 1 point
23 Alan wins this game!
```

4.1 Aufgabe

Implementiere eine Funktion `greetPlayerAskName`. Der Funktion wird eine Spielernummer als Integer übergeben. Sie fordert den Spieler dazu auf seinen Namen an der Konsole einzugeben. Der Name wird anschließend zurückgegeben.

Siehe Zeile 1 und Zeile 3 im Beispiel.

Nutze `greetPlayerAskNameHelper`!

4.2 Aufgabe

Schreibe eine Funktion `ask_rounds`, die fragt wie viele Runden gespielt werden sollen. Diese Eingabe wird als Integer zurückgegeben.

Siehe Zeile 5 im Beispiel.

4.3 Aufgabe

Implementiere eine Funktion `greetPlayerAskChoice`. Der Funktion wird eine Spielername als String übergeben. Sie fordert die Spielerin dazu auf Schere, Stein oder Papier einzugeben. Die Auswahl wird als String zurückgegeben.

Siehe Zeile 8, 10, 16 und 18.

Nutze `greetPlayerAskChoiceHelper`

4.4 Aufgabe

Implementiere eine Funktion `playOneRound`. Dieser Funktion werden die aktuelle Rundenzahl und die Zahl der Runden, die gespielt werden, übergeben. Sie gibt diese beiden Informationen in der Konsole aus. Der Funktion werden auch die Namen der Spieler übergeben. Sie spricht die Spieler mit ihren Namen an und bittet sie Schere, Stein oder Papier einzugeben. Das Ergebnis dieser Runde wird anschließend ausgegeben. Hierbei wird der Name des Gewinners genutzt.

Es wird auch zurückgegeben, wer gewonnen hat. Die einzig möglichen Rückgabewerte sind `"1"`, `"2"` und `"Nobody"`.

Aufrufe der Funktion sind in den Zeilen 7 - 12 und 15 - 20 zu sehen.

Nutze `showRoundNumber`, `greetPlayerAskChoice`, `computeRoundWinner` und `showRoundWinnerWithName`!

4.5 Aufgabe

Implementiere eine Funktion `playRPSHelper`. Diese Funktion werden die Anzahl der Runden, die gespielt werden und die Namen der Spieler übergeben. Sie spielt das Spiel unter diesen Bedingungen. Nachdem alle Runden gespielt wurden, wird das Endergebnis ausgegeben.

Ein Aufruf der Funktion ist ab Zeile 7 zu sehen.

Nutze `playOneRound`, `computeNewPoints`, `showPlayersPoints` und `showGameWinnerWithName`!

4.6 Aufgabe

Implementiere eine Funktion `playRPS`. Diese Funktion fragt zwei Spieler nach ihren Namen. Anschließend fragt sie wie viele Runden gespielt werden sollen. Danach wird das Spiel mit diesen Einstellungen durchgeführt.

Nutze `greetPlayerAskName`, `aksRounds` und `playRPSHelper`!

4.7 Aufgabe

Nutze das Kontrollzeichen `"\n"` im Code um bei der Ausgabe Zeilen zu überspringen.

5 Verbesserungen

5.1 Aufgabe

Implementiere eine Funktion `getCorrectInput`, die den Benutzer solange auffordert etwas einzugeben, bis die Eingabe `rock`, `paper` oder `scissors` ist. Diese Eingabe wird dann zurückgegeben.

5.2 Aufgabe

Implementiere eine Funktion `getCorrectRoundCounter`, die zwei Zahlen als Integer übergeben bekommt. Die Funktion fordert die Benutzerin auf eine Rundenzahl zwischen diesen beiden Zahlen einzugeben und fragt solange nach, bis die Eingabe korrekt war. "'

5.3 Aufgabe

Bau diese Funktionen in das vollständige Programm ein!