**Dokumentation Bash-Skript 20 gewinnt**

Graf, Lauper, Wenninger, Wilhelm

Ein Bild, das Logo enthält.

Automatisch generierte Beschreibung

**Inhaltsverzeichnis**

[**Einleitung** 3](#_Toc132227027)

[**Repository** 3](#_Toc132227028)

[**Spielregeln** 3](#_Toc132227029)

[**Auftrag** 3](#_Toc132227030)

[**Planung** 3](#_Toc132227031)

[**Arbeitsaufteilung** 3](#_Toc132227032)

[**Struktogramm** 4](#_Toc132227033)

[**Umsetzung** 4](#_Toc132227034)

[**Basis (Keine Features)** 4](#_Toc132227035)

[**Feature: Farbige Texte** 6](#_Toc132227036)

[**Feature: Username für Spieler** 6](#_Toc132227037)

[**Feature: Verschiedene Modis (PvP, PvE, PC vs PC und Tutorial)** 6](#_Toc132227038)

[**Feature: PC vs PC** 7](#_Toc132227039)

[**Feature: Tutorial** 8](#_Toc132227040)

[**Feature: Menü** 8](#_Toc132227041)

[**Was hätten wir am Skript besser machen können?** 8](#_Toc132227042)

[**Reflexion** 9](#_Toc132227043)

# **Einleitung**

## **Repository**

Link für das Repository: <https://github.com/LivioGraf/20Gewinnt>

## **Spielregeln**

Das Spiel können zwei Spieler spielen. Gewonnen hat der Spieler der zuerst 20 erreicht. Die Spieler können nacheinander eins oder zwei eingeben. Der Spiel Stand erhöht sich immer mit der Eingabe des Spielers.

## **Auftrag**

Erstellen Sie ein Bash-Skript, welches das Spiel umsetzt. Dokumentieren Sie Ihr Ergebnis und nutzen Sie die folgende Struktur dazu: Aufgabenstellung, Ihre Idee dargestellt durch ein Struktogramm, Umsetzung mit Erklärungen und Reflexion.

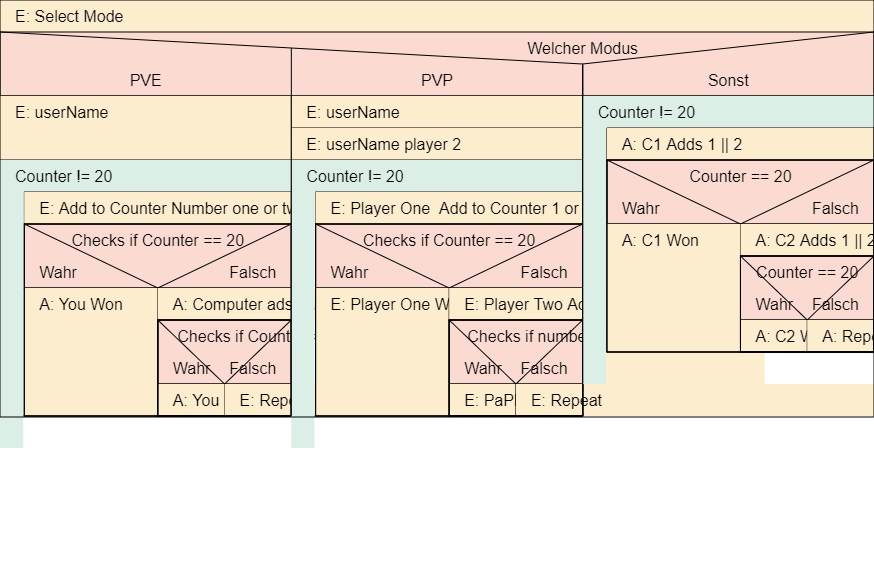
# **Planung**

## **Arbeitsaufteilung**

Bei der Aufteilung haben wir darauf geachtet, dass alle gleich viel machen müssen. Falls jemand mit seinem Teil fertig war, konnte er die andern noch unterstützen. So hatten wir alle stets genügend zu tun und konnten unser Projekt so reibungslos fertig stellen.

|  |  |
| --- | --- |
| **Mitglieder** | **Aufträge** |
| Marc | Struktogramm |
| Noah | Umsetzung |
| Kai | Dokumentation, Planung |
| Livio | Dokumentation, Umsetzung |

## **Struktogramm**



# **Umsetzung**

Das Spiel habe wir selbst geschrieben wir haben kein Code verwendet, der von einer AI generiert wurde. Auch die Features habe wir selbst geschrieben. Bei Problemstellungen oder Fehlermeldungen, die wir nicht verstanden haben, haben wir im Internet nachgeschaut und in Ausnamefällen haben wir Chat GPT gefragt, wieso das Skript einen Fehler verursacht. Wir haben aber Chat GPT nicht dazu verwendet, um das Skript für uns zu schreiben.

## **Basis (Keine Features)**

Das Basis Skript also das Spiel ohne Features hat 50 Zeilen.

Das spiel beginnt mit einigen Ausgaben, die den Spieler begrüssen und das Spiel kurz erläutern.  
Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Nach der Begrüssung kann der Spieler seinen Namen eingeben das haben wir mit einem echo gemacht und dann die read Funktion verwendet, um den Userinput zu speichern.



Im Code folgt dann eine Variable, die wir gameCount genannt haben, die wird dazu verwendet wird, um zu überprüfen, ob 20 schon erreicht wurde. Der Startwert haben wir auf null gesetzt da das Spiel bei 0 beginnt und die Spieler dann den Count um eins oder zwei erhöhen können.



Das eigentliche game spielt sich im folgenden Game loop ab im dem wir überprüfen, ob gameCount kleiner als 20 ist.  


Am Anfang des Loops kann der Spieler sein Eingabe machen, diese Eingabe wird dann so validiert das wenn sie nicht eins oder zwei ist, wird ein Fehler ausgegeben. Die Validierung haben wir mit einem Regex gemacht.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Nach dem der Input validiert ist wird überprüft, ob der Spieler gewonnen hat. Um das zu Prüfen haben wir eine Funktion geschrieben, die sich checkIfWon nennt. Der Funktion muss man drei Parameter übergeben den Userinput, den gameCount und gewonnen Nachricht, falls der Spieler gewonnen hat. Die Funktion funktioniert so das überprüft wird, ob der gameCount plus die Usereingabe grösser oder gleich 20 ist. Wenn das der Fall ist, wird die Gewonnen Nachricht und der gameCount ausgegeben. Das Skript wird dann auch abgebrochen mit exit.



Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Falls der Spieler nach dem Überprüfen, ob er gewonnen hat, nicht gewonnen hat wird der gameCount mit der Usereingabe addiert.



Nachdem der Spieler seine Eingabe gemacht hat, ist der Computer am Zug. Zuerst wird eine Ausgabe gemacht in der gesagt wird das jetzt der Computer am Zug ist.



Um die Eingabe des Computers zu simulieren haben wir eine zufällige Zahl zwischen eins und zwei generiert. Diese generierte Zahl haben wir dann in ausgegeben.

Ein Bild, das Text, Uhr, Gerät, Messgerät enthält.

Automatisch generierte Beschreibung

Den Input des Computers mussten wir nicht validieren da er nur die Möglichkeit hat valide Inputs zu machen. Wir mussten aber wie beim Spieler überprüfen, ob der Computer gewonnen hat, um das zu machen haben wir wieder die checkIfWon Funktion verwendet.



Wenn der Computer nicht gewonnen hat, dann wird der gameCount so wie beim Spieler addiert mit der Eingabe des Computers.



Am Ende wird noch der gameCount ausgegeben.



## **Feature: Farbige Texte**

Um die Texte farbig zu machen haben wir immer echo -e verwendet und verschieden ANSI Escape Codes für unterschiedliche Farben.

Beispiele:







## **Feature: Username für Spieler**

Um den Usernamen zu speichern haben wir eine Ausgabe gemacht in der steht das der Spieler hier seinen Usernamen eingeben soll. Mit der read Funktion haben wir diesen dann in einer Variable gespeichert. Beim PvP Modus sind wir gleich vorgegangen.



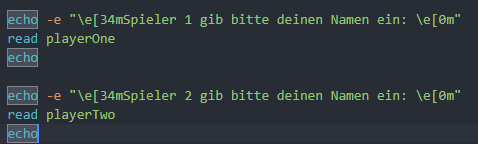
Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

## **Feature: Verschiedene Modis (PvP, PvE, PC vs PC und Tutorial)**

Wir haben drei verschiedene Modis und ein Tutorial implementiert. Den PvE Modus werden wir hier nicht erklären da dieser unsere Basis Skript ist (hier dokumentiert: [PvE](#_Basis_(Keine_Features))). Der Code für den PvP Modus ähnelt sehr dem code für den PvE Modus. Sie unterscheiden sich nur an zwei Stellen ansonst ist alles gleich. Um zu überprüfen, ob ein Spieler gewonnen hat, wird auch mit der checkIfWon Funktion gemacht wie im PvE Modus.

Der erste unterschied ist das am Anfang des Spieles nicht nur ein Spieler seinen Namen eingeben muss sondern beide Spieler.



Der zweite Unterschied ist das beim Spieler zwei die Input Validierung etwas anders funktioniert. Die Validierung der Eingabe wird in einem while true loop gemacht da der Spieler so lange eine Eingabe machen muss, bis sie valid ist. Im Loop wird mit einem if else Statement und einem Regex überprüft, ob die Eingabe Valid ist. Wenn sie nicht Valid ist, bekommt der Spieler eine Fehlermeldung und im else Fall wir der Loop abgebrochen.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

## **Feature: PC vs PC**

Die PC-Simulation beginnt mit der Initialisierung und Deklaration von den Variablen computerOne und computerTwo. Die Variablen werden mit den Werten computerOne und computerTwo initialisiert. Diese Variablen brauchen wir später für ausgaben damit die Ausgaben nicht Hartkodiert sind.



Auch hier haben wir wieder eine gameCount Variable gemacht.



Der Game Loop ähnelt sehr den anderen Modis. In der while Bedingung wird auch überprüft, ob der gameCount kleiner als 20 ist.



Dann wird eine Ausgabe gemacht das der computerOne am Zug ist. Die Eingabe ist eine zufällig generierte Zahl zwischen eins und zwei. Die generierte Zahl wird dann ausgegeben.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Danach wird mit der checkIfWon Funktion überprüft, ob computerOne gewonnen hat. Wenn er nicht gewonnen hat, wird der gameCount mit der Eingabe erhöht. Der gameCount wird dann ausgegeben.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Der gleiche Vorgang wiederholt sich dann für den computerTwo.

## **Feature: Tutorial**

Im Tutorial wird mit Ausgaben und einem kurzen Beispiel das Spiel erklärt. Am Ende des Tutorials hat der User die Möglichkeit mit der Eingabe eins wieder ins Menü zu gelangen und mit der Eingabe zwei wird das Skript abgebrochen.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

## **Feature: Menü**

Im Menü kann der User auswählen welcher Modus er Spielen will. Das Menü startet mit einem while true Loop in dem der User am Anfang mit eins, zwei, drei oder vier zwischen den Modis auswählen kann. Wenn die Eingabe nicht eins, zwei, drei oder vier ist wird ein Fehler ausgegeben.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Mit if elif werden je nach Eingabe die verschiedenen Modis gestartet.









## **Was hätten wir am Skript besser machen können?**

Wir hätten die checkIfWon Funktion anders implementieren können. Anstelle von einem exit am ende der Funktion könnte man einen Boolean zurückgeben und beim Aufruf der Funktion mit einem if Statement auf die Rückgabe reagieren und so das Skript steuern.

Da der Code für die Inputvalidierung immer der gleiche ist hätten wir diesen Code auch in eine Funktion auslagern können.

# **Reflexion**

Abschliessend können wir sagen das wir bei diesem Projekt sehr viel neues gelernt haben. Konkret wissen wir jetzt, wie man ein Bash-Skript von Grund aus aufsetzt und darauf aufbauen kann, wie zum Beispiel die zusätzlichen Features (pvp Modus, Farbe vom Spieler ändern usw.) Wir alle hatten vorher nur sehr wenig Erfahrung im Bereich Scripting, was sich nun geändert hat. Als Gruppe konnten wir hervorragend zusammenarbeiten und wussten auch wie man mit Konflikten umgeht und diese so schnell wie möglich lösen kann. Falls jemand Hilfe benötigte oder nicht mehr weiterkam, haben ihn die andern aus dem Team immer unterstützt und geholfen.