Aufgabe 1: Hopsitexte

Team-ID: 00433

Team-Name: Was?

Bearbeiter dieser Aufgabe:
Mathieu de Borman

1. Dezember 2025

Inhaltsverzeichnis

1 Lösungsidee

Es gibt ein Texteingabefeld, in dem automatisch bei jeder Änderung die Farben passend eingetragen werden.

1.1 Markieren des Texts

Zuerst wird jede Position erfasst, bei der jemand landet, dann wird diese je nach Fall passend aufgetragen.

1.2 Laufzeitanalyse

Die Laufzeit hängt von einem Parameter ab, nämlich der Anzahl der Buchstaben n. Das Programm hat eine Laufzeitkomplexität von $O(n)$ bei einer Speicherkomplexität von $O(n)$.

2 Umsetzung

Die Lösungsidee ist in Python implementiert und verwendet tkinter um das Dialogfeld zu erstellen.

2.1 Markieren der Buchstaben

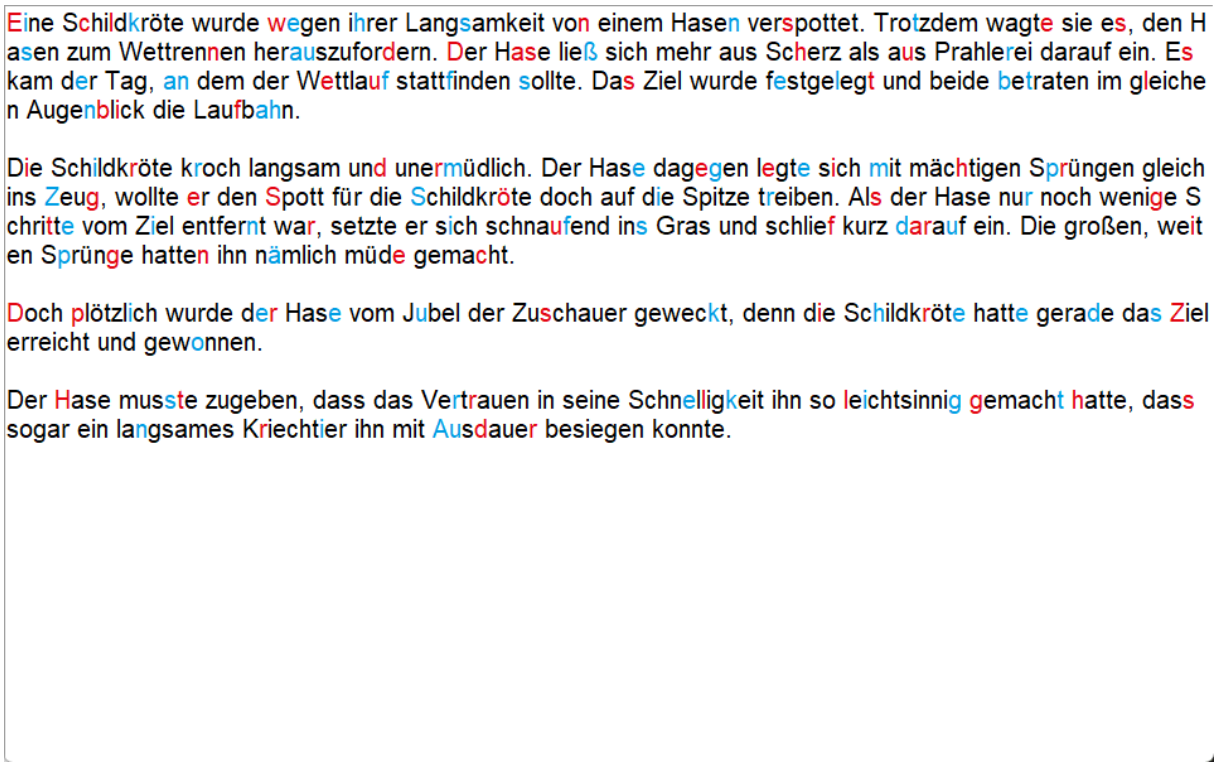
Zuerst werden alle irrelevanten Buchstaben herausgefiltert und der Rest zu Kleinbuchstaben umgewandelt. Anschließend wird durch jeden Sprungpunkt jedes Spielers iteriert, wobei sich diese Punkte auch gemerkt werden. Danach wird durch jeden Buchstaben iteriert wobei jeder passend bemalt wird. Wenn beide auf einem Buchstaben landen, dann wird dieser Rot hinterlegt. Landet nur einer der Beiden auf einer Position, dann wird der Buchstabe selber passend gefärbt (Bela: rot, Amira: blau).

3 Beispiele

Wir rufen nun das Python-Programm mit den verschiedenen JWINF-Eingabedateien auf. Diese Dateien sind in demselben Ordner wie die Programmdatei. In dem Textfeld kann man dann den Text schreiben. Der Text wird automatisch markiert.

3.1 Lösungen

3.1.1 hopsen1.txt



3.1.2 hopsen2.txt

```
Ein Federchen flog durch das Land;
Ein Nilpferd schlummerte im Sand.

Die Feder sprach: "Ich will es wecken!"
Sie liebte, andere zu necken.

Aufs Nilpferd setzte sich die Feder
Und streichelte sein dickes Leder.

Das Nilpferd sperrte auf den Rachen
Und musste ungeheuer lachen.
```

3.1.3 hopsen3.txt

```
Koukonisi ist eine kleine, nicht bewohnte griechische Insel im Golf von Moudros der Insel Limnos. Diese Insel
liegt nördlich von Moudros und gehört zu dessen Gemeindebezirk. Koukonisi ist über eine befestigte Straße v
on der etwa 400m entfernten Küste zu erreichen.
```

3.1.4 hopsen4.txt

128 Zeichen umfasst das ASCII-System und stellt damit eine simple, aber effektive Möglichkeit dar, Texte di gital zu codieren. Jeder Buchstabe des lateinischen Alphabets sowie grundlegende Satzzeichen und Zahlen si nd darin enthalten. Diese Beschränkung auf 128 Zeichen machte ASCII besonders für frühe Computer attrakt iv, da Speicherplatz sehr knapp war. Auch wenn moderne Systeme komplexere Codierungen nutzen, bleibt A SCII in vielen Bereichen relevant.

4 Quellcode

```
#Bibliotheken für das Dialogfeld
import tkinter as tk
import re

def main():
    #definiere die buchtsaben, die gezählt werden sollen
    letters = ["a","b","c","d","e","f","g","h","i","j","k","l","m","n",
    "o","p","q","r","s","t","u","v","w","x","y","z","ä","ö","ü","ß"]
    #öffne das Dialogfeld auf halber Bildschirmgröße
    window = tk.Tk()
    window.title("A1")
    width = window.winfo_screenwidth() >> 1
    height = window.winfo_screenheight() >> 1
    window.geometry("%dx%d" % (width, height))
    #Füge ein Textfeld hinzu
    inputfield = tk.Text(window, font=("Arial", 15, ""))
    inputfield.pack(expand=True, fill=tk.BOTH)
    #funktion um den Text zu markieren
    def analyse(event):
        #setze den Text zurück
        inputfield.edit_modified(False)
        for tag in inputfield.tag_names():
            inputfield.tag_delete(tag)
        #deklariere die Farben, die die Buchstaben haben sollen
        inputfield.tag_config("playerOne", foreground="#e30613")
        inputfield.tag_config("playerTwo", foreground="#009ee4")
        inputfield.tag_config("colision", background="red")
        #Standartisiere die Eingabe auf nur Kleinbuchstaben
        text = inputfield.get("1.0", "end-1c")
```

```
        text = text.lower()
        textLetters = re.sub(r"[^a-zA-ZäöüÄÖÜß]", "", text)
        #Setze die Positionen der Spieler auf ihre jeweiligen Startfelder
        playerOnePositions = [0]
        playerTwoPositions = [1]
        #Merke dir jede Position, bei der der erste Spieler vorbeikommt
        i = 0
        while i < len(textLetters):
            i += letters.index(textLetters[i]) + 1
            playerOnePositions.append(i)
        #Merke dir jede Position, bei der der zweite Spieler vorbeikommt
        i = 1
        while i < len(textLetters):
            i += letters.index(textLetters[i]) + 1
            playerTwoPositions.append(i)
        #gehe an den anfang des Texts
        letterIndex = 0
        letter = 0
        line = 1
        #für jeden Buchstaben
        for i in range(len(text)):
            #Falls dieser für die Färbung relevant ist
            if text[i] in letters:
                #falls beide Spieler auf derselben Position landen, dann hinterlege diese Rot
                if letterIndex in playerTwoPositions and letterIndex in playerOnePositions:
                    inputfield.tag_add("colision", f"{line}.{letter}", f"{line}.{letter+1}")
                #Wenn nur Spieler eins drauf landet, färbe es Rot
                elif letterIndex in playerOnePositions:
                    inputfield.tag_add("playerOne", f"{line}.{letter}", f"{line}.{letter+1}")
                #Wenn nur Spieler zwei drauf landet, färbe es Blau
                elif letterIndex in playerTwoPositions:
                    inputfield.tag_add("playerTwo", f"{line}.{letter}", f"{line}.{letter+1}")
                #Ein weiterer Buchstabe wurde gefärbt
                letterIndex += 1
            #Die zu überprüfende stelle ist jetzt eins weiter
            #Entweder in der nächsten Stelle oder Zeile
            if text[i] != "\n":
                letter += 1
            if text[i] == "\n":
                letter = 0
                line += 1
    #Bei änderung des Texts bemale es neu
    inputfield.bind("<<Modified>>", analyse)
    window.mainloop()
#Führe den gesamten code aus
if __name__ == "__main__":
    main()
```