

1. Ein Systemaufruf ist eine Routine, die es einer Benutzeranwendung ermöglicht, Aktionen anzufordern, die spezielle Berechtigungen erfordern. Das Hinzufügen von Systemaufrufen ist eine von mehreren Möglichkeiten, die vom Kernel bereitgestellten Funktionen zu erweitern.

Ein Systemaufruf unterscheidet sich in mehreren wichtigen Punkten von einer Benutzerfunktion.

Ein Systemaufruf hat mehr Privilegien als eine normale Subroutine. Ein Systemaufruf wird mit Kernel-Mode-Berechtigung in der Kernel-Schutzdomäne ausgeführt.

Systemaufrufcode und -daten befinden sich im globalen Kernelspeicher.

Systemaufrufroutinen können Kernelprozesse erstellen und verwenden, um eine asynchrone Verarbeitung durchzuführen.

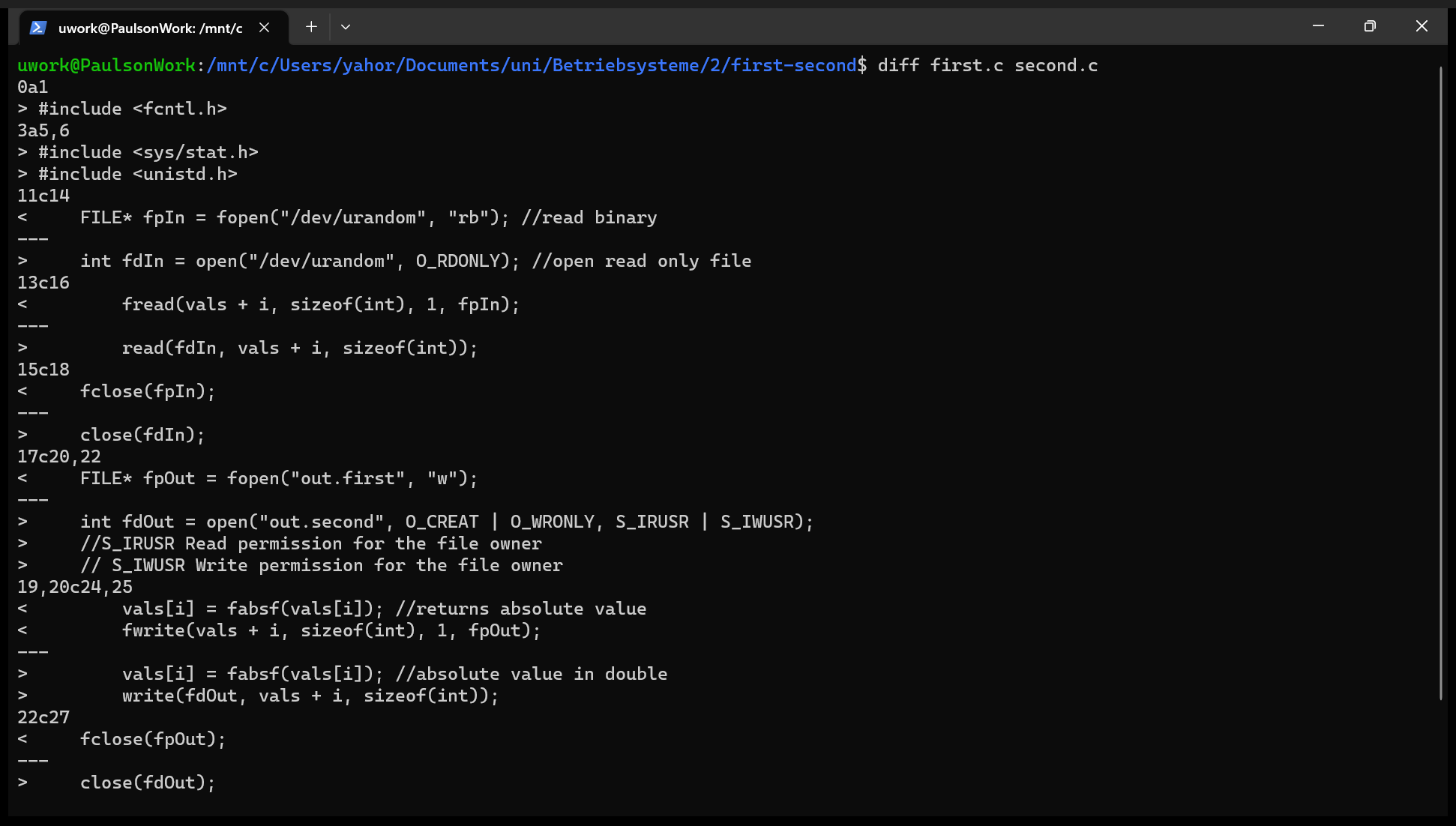
Systemaufrufe können keine gemeinsam genutzten Bibliotheken oder Symbole verwenden, die nicht in der Kernel-Schutzdomäne gefunden werden.

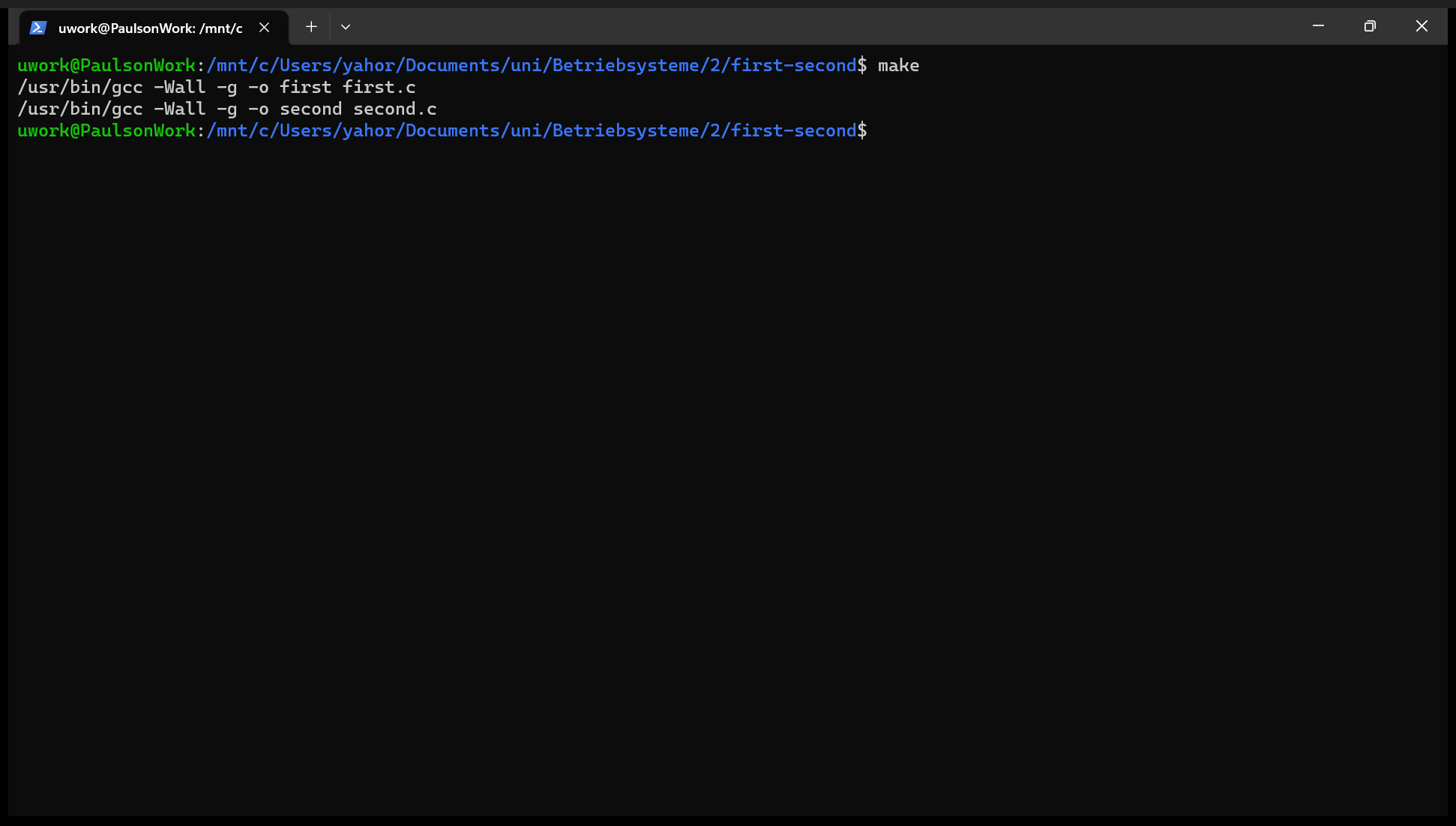
1. fork() process erstellen

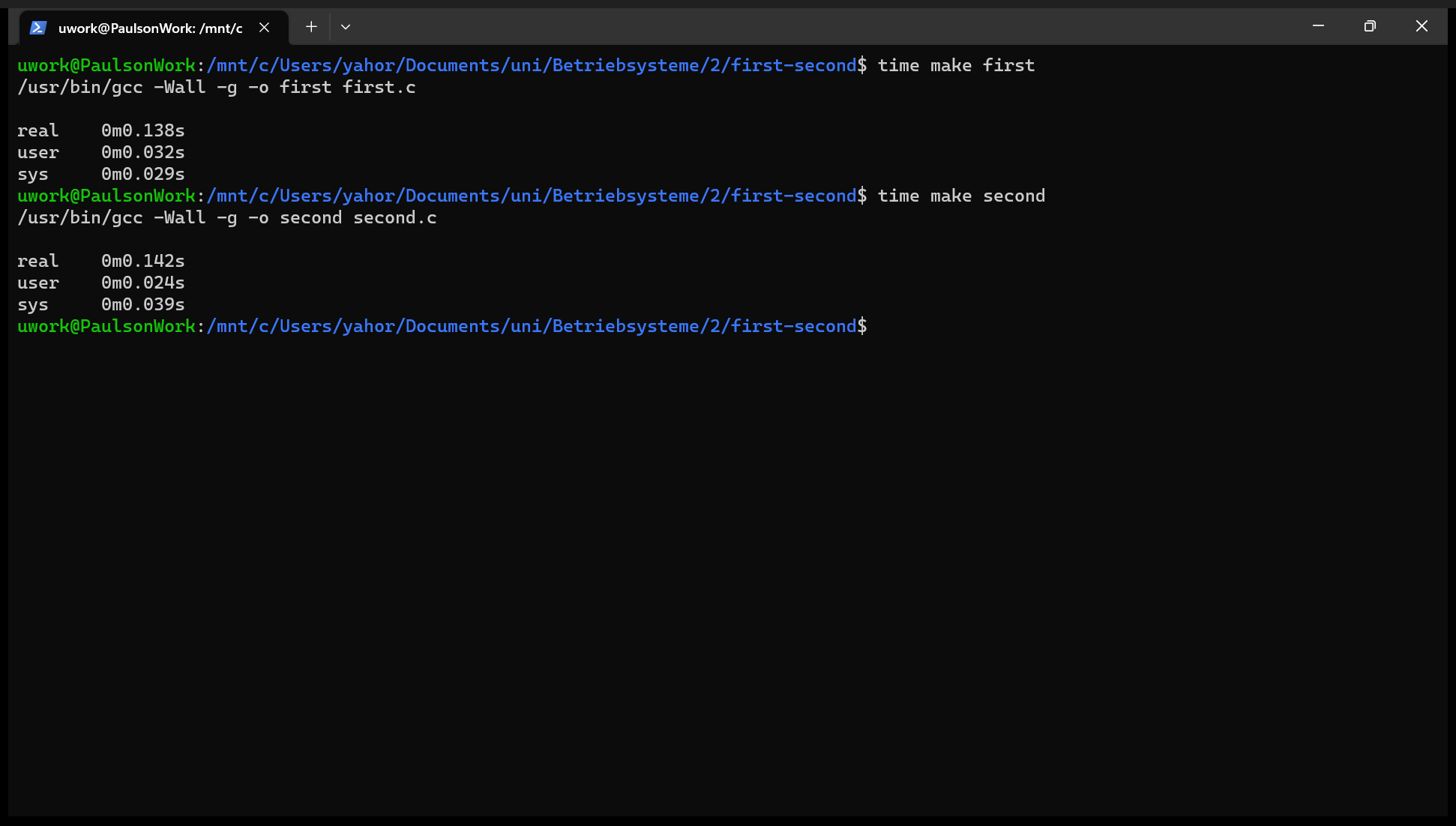
exit() „sofort“ den Prozess beenden

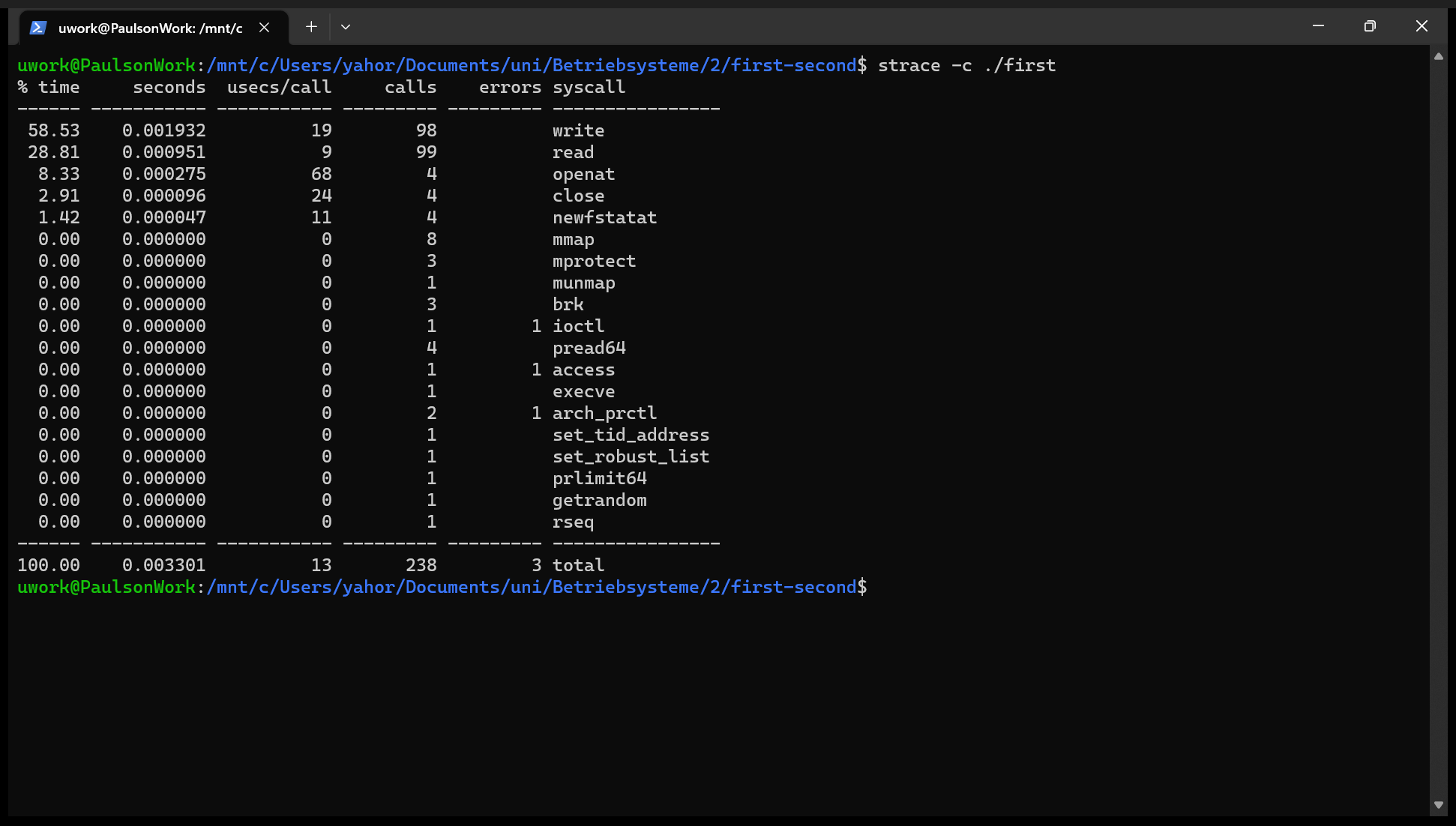
wait() blockiert den aufrufenden Prozess, bis einer seiner untergeordneten Prozesse beendet wird oder ein Signal empfangen wird

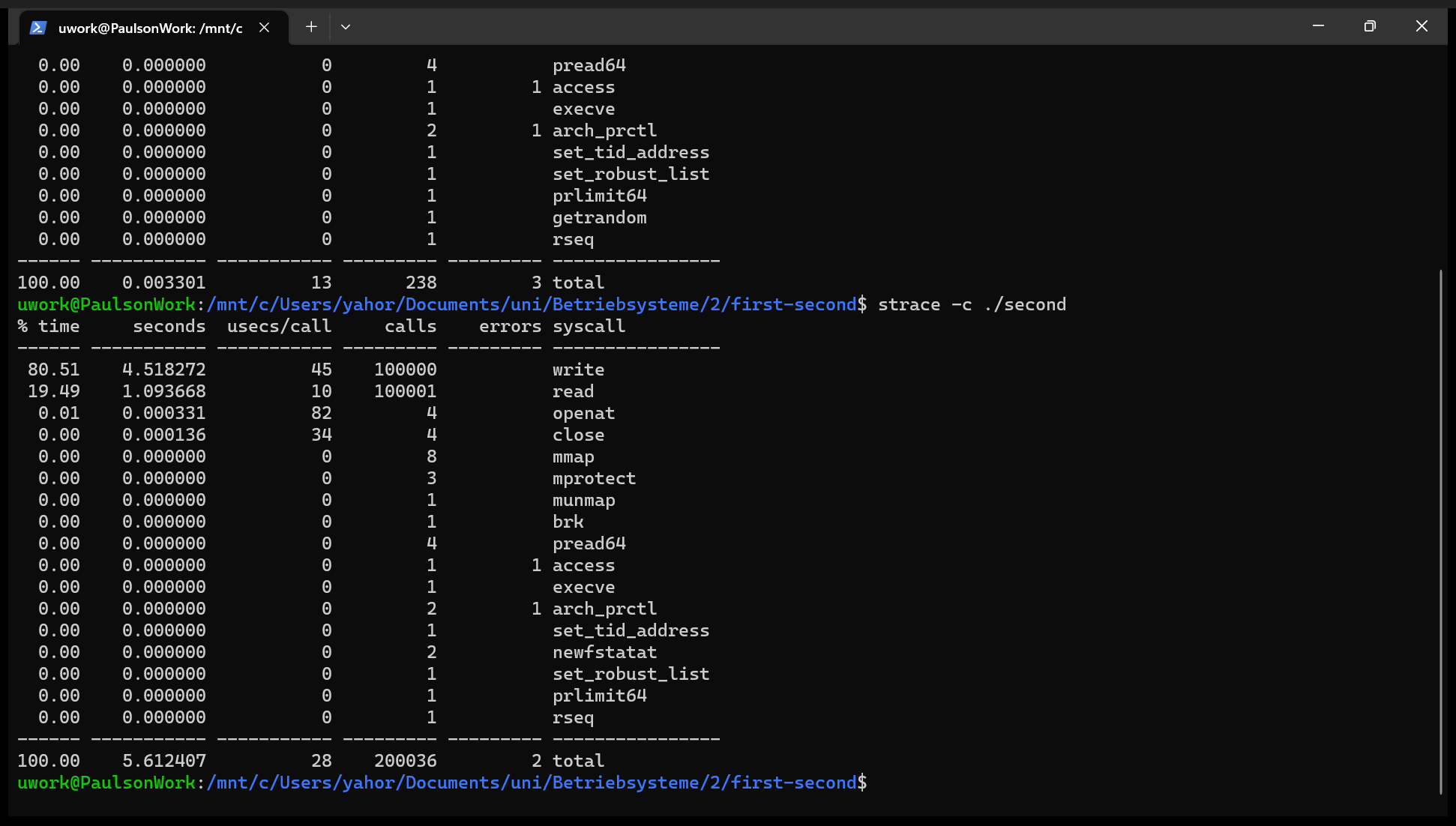
1. Der Systemaufruf kill() kann verwendet werden, um ein beliebiges Signal an einen beliebigen Prozess (Gruppe) zu senden.

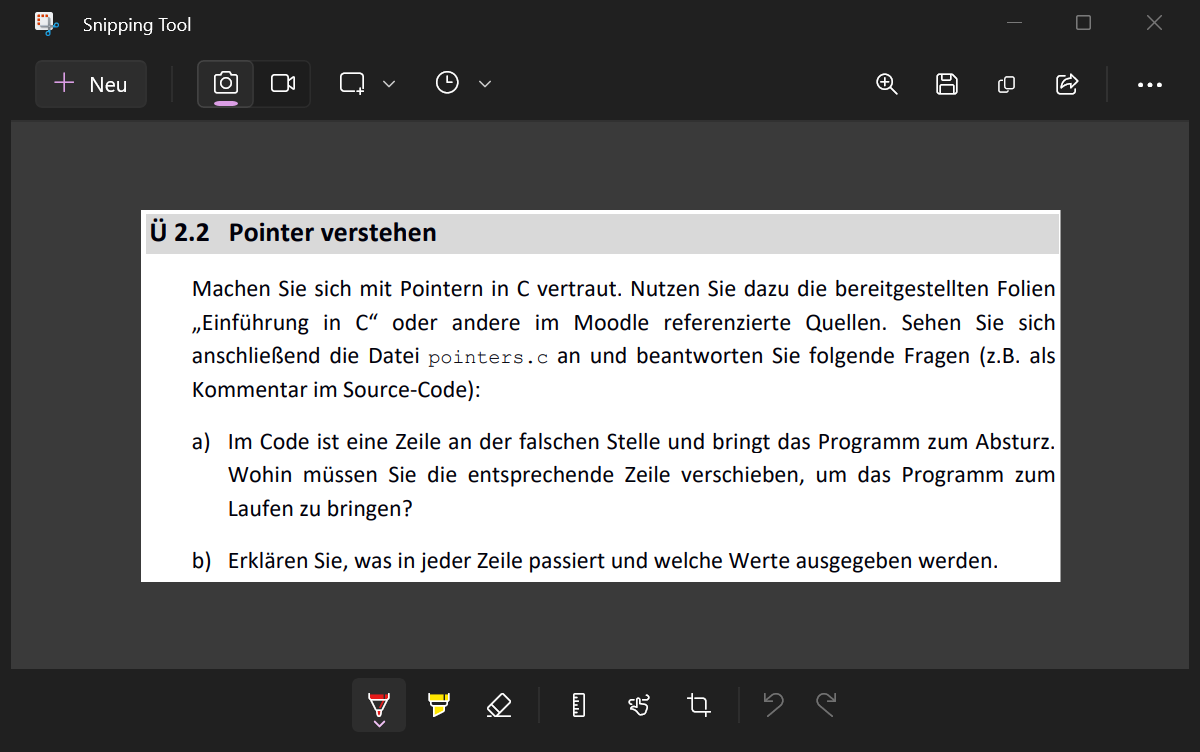


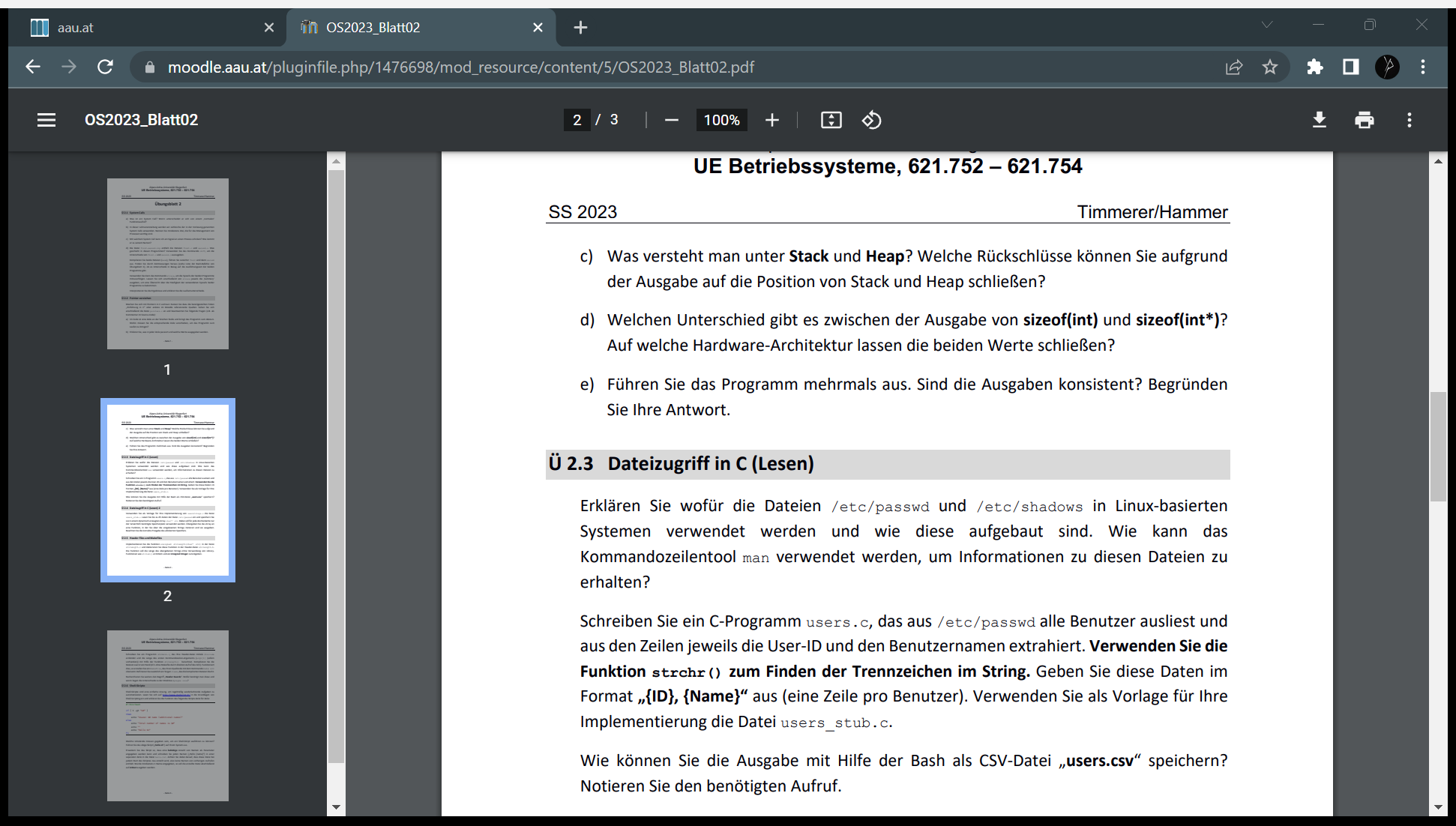


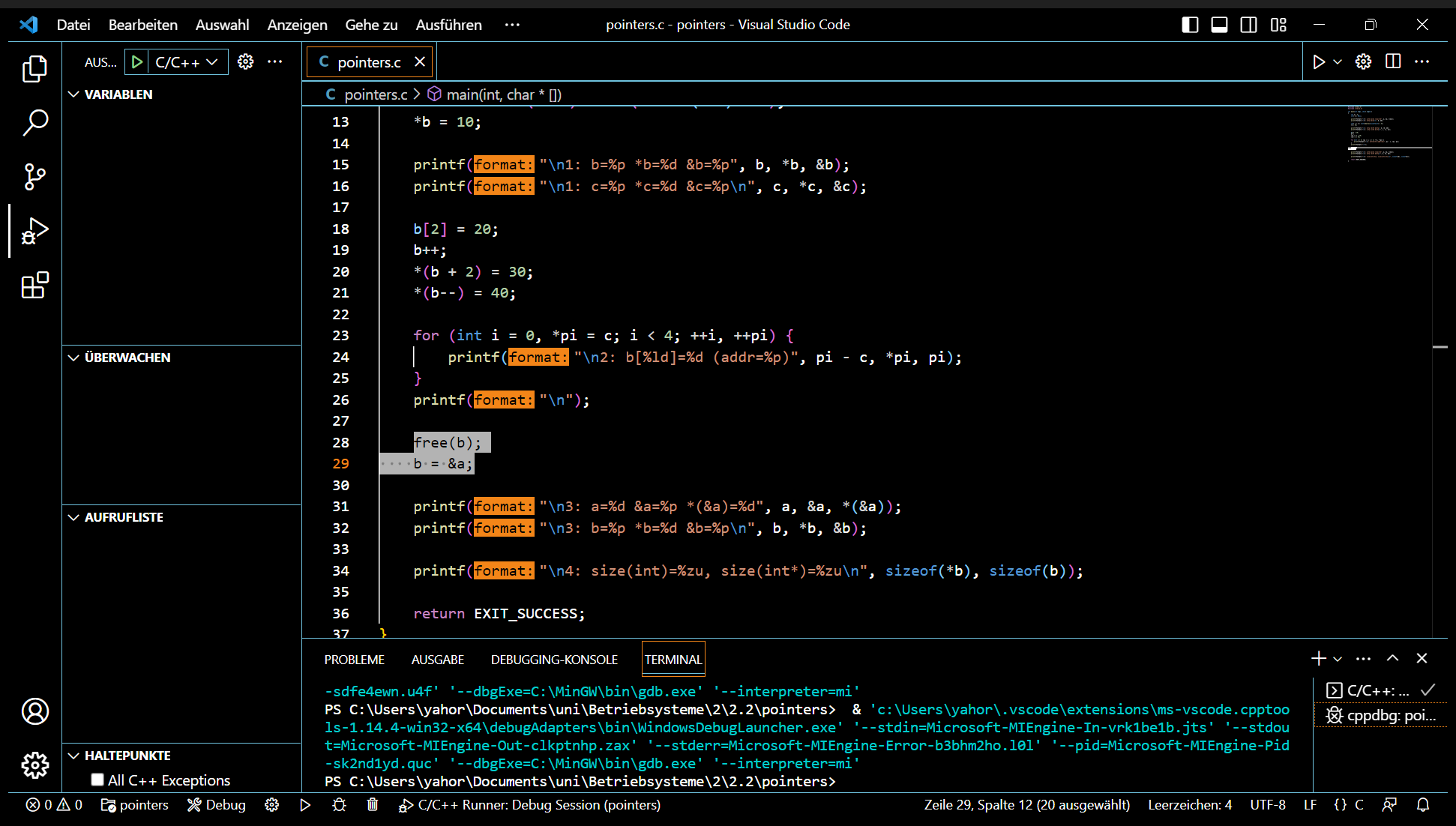


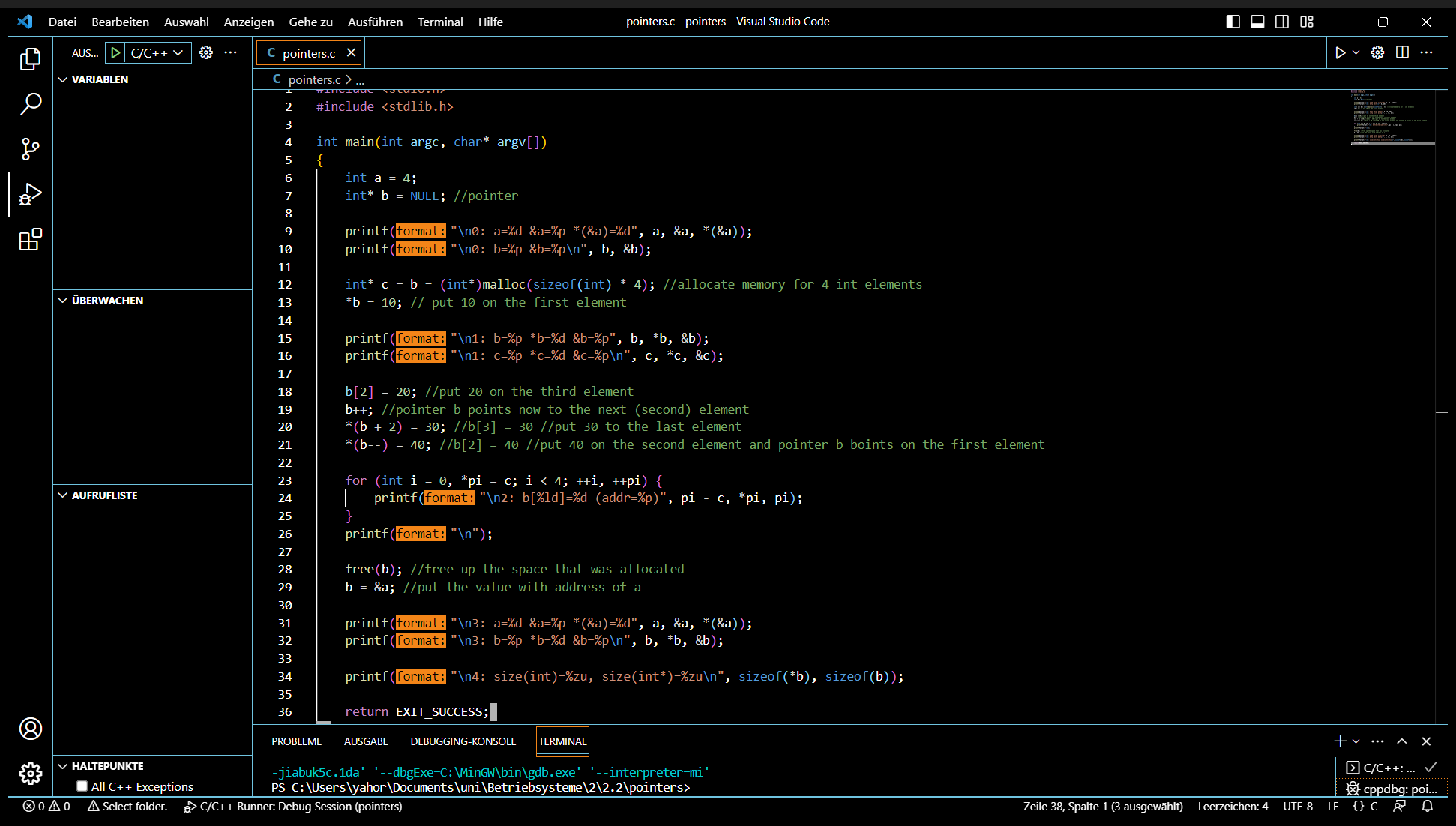










1. 
2. Ein **Stack** ist eine lineare Datenstruktur, eine Sammlung von Elementen des gleichen Typs. In einem Stack geschieht das Einfügen und Löschen von Elementen nur an einem Endpunkt. Das Verhalten eines Stapels wird als „Last In, First Out“ (LIFO) beschrieben.

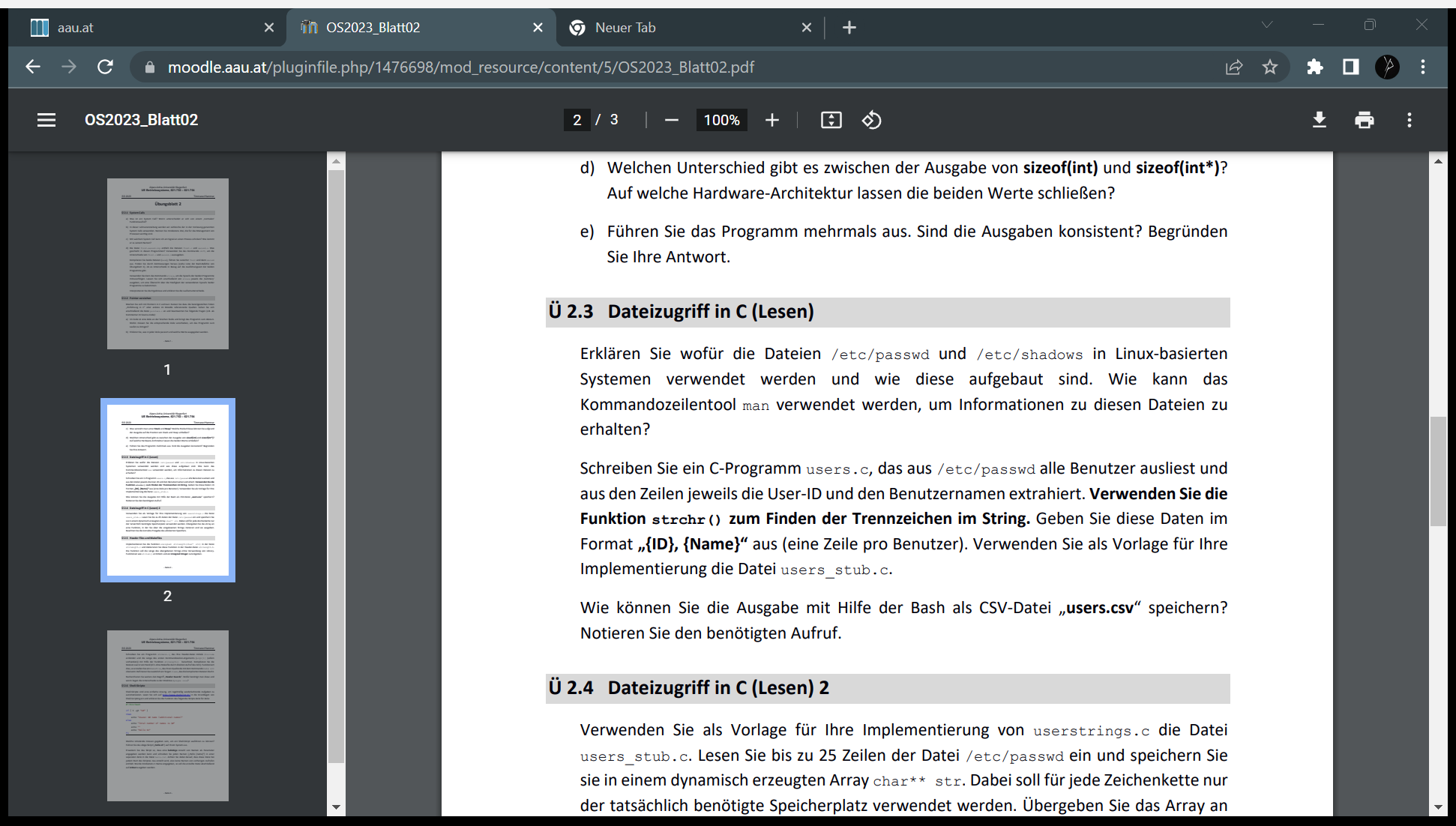
**Heap** ist ein Bereich des vorreservierten Computerhauptspeichers (Speicher), den ein Programmprozess verwenden kann, um Daten in einer variablen Menge zu speichern, die nicht bekannt ist, bis das Programm ausgeführt wird.

1. **int** bedeutet eine Variable, deren Datentyp Integer ist.

**sizeof(int)** gibt die Anzahl der Bytes zurück, die zum Speichern einer Ganzzahl verwendet werden.

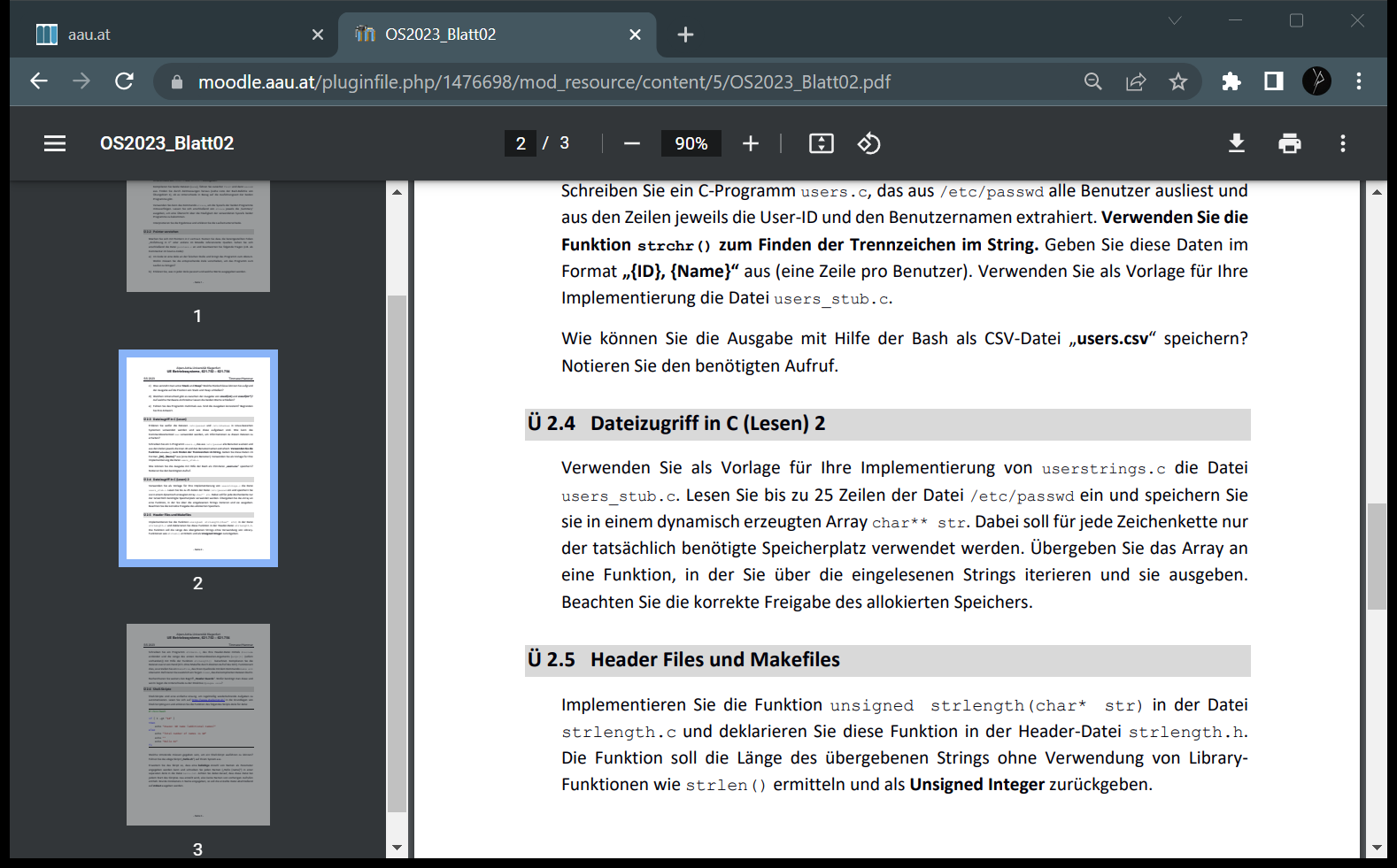
**int\*** bedeutet einen Zeiger auf eine Variable, deren Datentyp Integer ist.

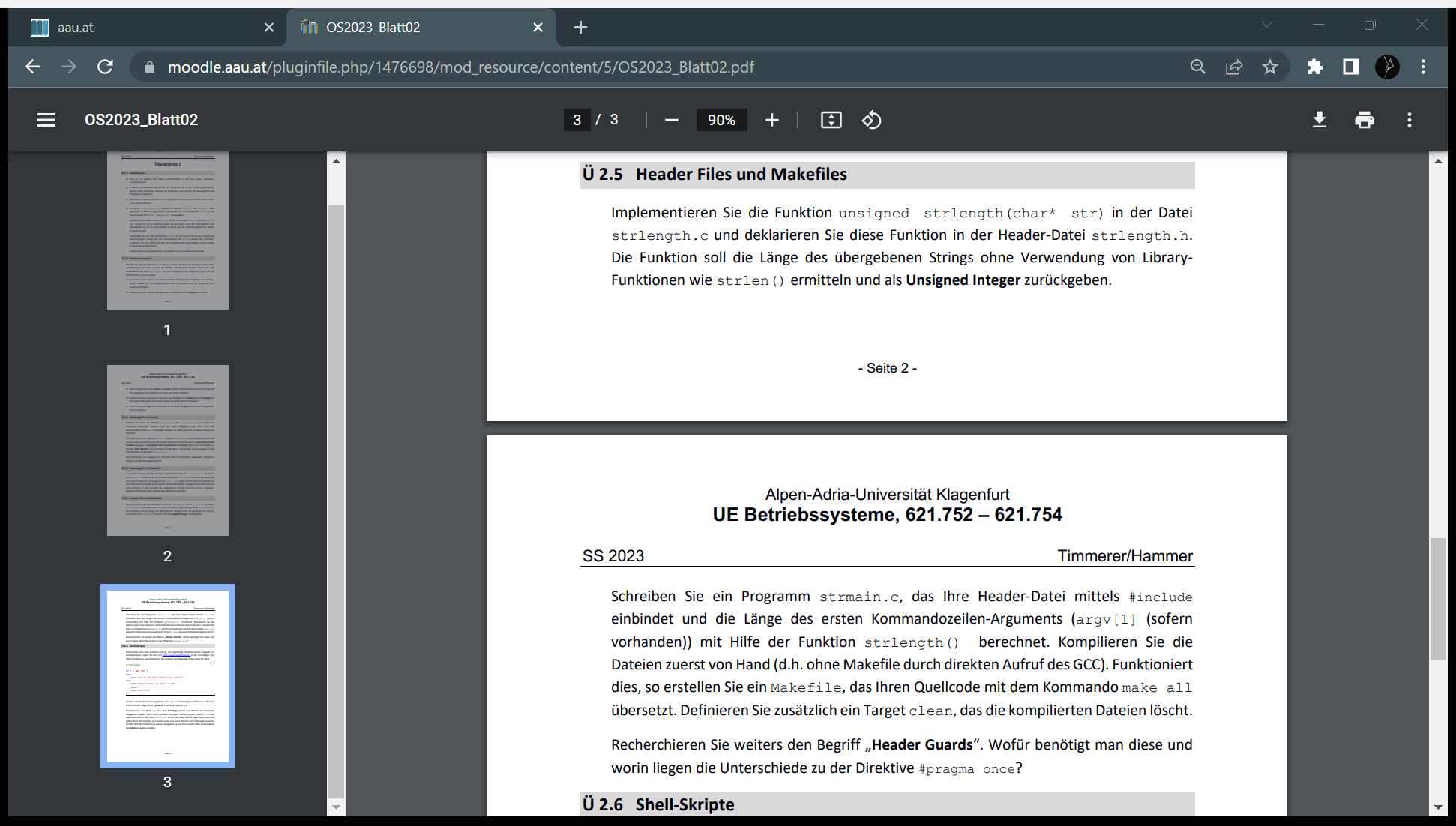
**sizeof(int\*)** gibt die Anzahl der Bytes zurück, die zum Speichern eines Zeigers verwendet werden.



Die Datei /etc/passwd speichert wichtige Informationen, die während der Anmeldung erforderlich sind. Mit anderen Worten, es speichert Benutzerkontoinformationen. Die /etc/passwd ist eine reine Textdatei. Es enthält eine Liste der Konten des Systems, die für jedes Konto einige nützliche Informationen wie Benutzer-ID, Gruppen-ID, Home-Verzeichnis, Shell und mehr enthält. Die Datei /etc/passwd sollte eine allgemeine Leseberechtigung haben, da viele Befehlsdienstprogramme sie verwenden, um Benutzer-IDs Benutzernamen zuzuordnen. Der Schreibzugriff auf /etc/passwd darf jedoch nur für das Superuser-/Root-Konto beschränkt werden.

/etc/shadow ist eine textbasierte Passwortdatei. Die Shadow-Datei speichert das gehashte Passphrase-Format (oder „Hash“) für das Linux-Benutzerkonto mit zusätzlichen Eigenschaften in Bezug auf das Benutzerpasswort. Auf diese Shadow-Datei kann nur der Root-Benutzer direkt zugreifen. Einige Befehle oder Programme (z. B. su, passwd und andere) mit eindeutiger SetUID (SUID) sind jedoch darauf eingestellt, um ihnen Zugriff auf die Shadow-Datei zu gewähren.





Header Guards sollen sicherstellen, dass der Inhalt einer bestimmten Header-Datei nicht mehr als einmal in eine einzelne Datei kopiert wird, um doppelte Definitionen zu vermeiden.

#pragma once ist optimiert, um die Verwendung desPräprozessors und das Öffnen von Dateien zu reduzieren, wodurch es schneller ist als Guards einzuschließen.

