Ariane 5 - eine Rakete stürzt wegen eines Programmierfehlers ab.

"Tauchen wir ein in eine faszinierende, aber auch erschreckende Geschichte aus der Welt der Raumfahrt. Es war 1996, als die Ariane 5, eine Rakete, die für Millionen von Euro entwickelt worden war, nur 40 Sekunden nach dem Start explodierte!

Die Ariane 5 war ein unglaublich teures und komplexes Projekt, entwickelt, um Satelliten ins All zu bringen. Doch bei ihrem ersten Flug ging etwas schief. Kurz nach dem Start explodierte die Rakete und zerstörte alle vier an Bord befindlichen Forschungssatelliten. Der finanzielle Schaden betrug rund 370 Millionen Dollar!

Schuld war ein Laufzeitfehler. Nach eingehender Untersuchung stellte sich heraus, dass ein Überlauffehler in der Software des Bordcomputers der Rakete für die Katastrophe verantwortlich war. Bei einem Überlauffehler versucht ein Programm, eine größere Zahl zu speichern, als es speichern kann. Im Fall der Ariane 5 versuchte das Programm, eine große 64-Bit Fließkommazahl in eine kleinere 16-Bit Ganzzahl umzuwandeln. Da die Geschwindigkeit der Ariane 5 höher ist als die der älteren Ariane 4, war die Zahl, die das Programm verarbeiten musste, zu groß und es kam zu einem Überlauffehler.

Ariane 5 explodiert ca. 40 Sek nach dem Start.

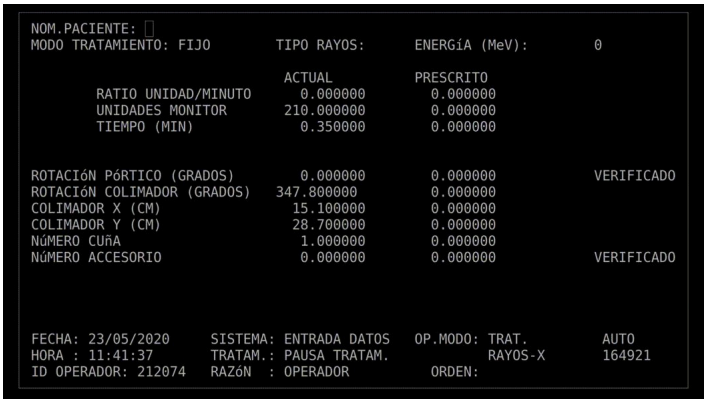
Da dieser Fehler erst auftrat, als das Programm bereits lief, wird er als Laufzeitfehler bezeichnet. Das fehlerhafte Programm lieferte dem Steuerungssystem der Rakete falsche Daten und brachte sie vom Kurs ab, was schließlich zur Explosion führte.

Diese Geschichte zeigt, wie wichtig es ist, Software sorgfältig zu testen und zu debuggen. Ein einziger unentdeckter Laufzeitfehler kann katastrophale Folgen haben. Deshalb ist es wichtig, dass wir als angehende Programmiererinnen und Programmierer lernen, wie man Fehler findet und behebt!

Therac 25 - Die Bedeutung des Debuggings: Eine (tödliche) Lektion aus der Geschichte

Wie wichtig diese Fehlersuche ist, zeigt ein historisches Beispiel: die Bestrahlungsmaschine Therac 25 aus den 1980er Jahren. Die Maschine wurde eingesetzt, um Krebspatienten mit Strahlentherapie zu behandeln. Doch dann passierte etwas Schlimmes: Einige Patienten erhielten eine viel zu hohe Strahlendosis - rund 100 Mal mehr als vorgesehen. Dieser Fehler führte zu schweren Verletzungen und in einigen Fällen sogar zum Tod der Patienten.

Warum ist das passiert? Die Antwort liegt in einem 'semantischen Fehler' in der Software des Geräts. Ein semantischer Fehler tritt auf, wenn ein Programm sich anders verhält als vom Programmierer beabsichtigt. In diesem Fall führte der semantische Fehler dazu, dass die Maschine manchmal eine viel zu hohe Strahlendosis abgab.



Therac 25 Input Screen

Die Programmierer wussten zunächst nicht, warum dieser Fehler auftrat. Sie mussten den Fehler in der Software 'debuggen', um herauszufinden, was schief gelaufen war. Das heißt, sie mussten Schritt für Schritt durch den Code gehen, um zu verstehen, warum das Programm nicht so funktionierte, wie sie es erwartet hatten.

Schließlich konnten sie den Fehler identifizieren und beheben. Dieser Fall zeigt, wie wichtig das Debugging bei der Softwareentwicklung ist. Er zeigt auch, wie schwerwiegend semantische Fehler sein können und warum wir als Programmierer immer darauf achten müssen, unseren Code sorgfältig zu testen und zu debuggen.

Text auf Grundlage des Seminar-Textes von Martin Pfeifer - Uni Koblenz.