ANLEITUNG

Allgemeine Tipps zum Programmieren und Lösen von Problemen

Programmieren kann gerade am Anfang überwältigend sein. Damit ihr Euch so bald wie möglich wohlfühlt in der Welt der Programmierung, sind hier ein paar Tipps, die den Einstieg erleichtern. Zunächst erhaltet Ihr einige allgemeine Tipps zur Lösung von Problemen, dann noch ein paar Hinweise, was Ihr machen könnt, wenn Euer Code nicht das tut, was er tun soll.

Gebt das Problem in eigenen Worten wieder

Der erste Schritt zur Lösung eines Problems besteht darin, genau zu verstehen, was das Problem ist. Wenn das Ziel nicht klar ist, weiß man nicht, wann man angekommen ist, und es ist kaum möglich zu planen, wie man dorthin kommt. Gebt daher das Problem in eigenen, einfachen Worten wieder. Erklärt das Problem schriftlich auf Papier oder jemandem mündlich. Achtet dabei darauf, dass Ihr folgende Fragen beantwortet:

- Was ist das Ziel?
- Welchen Input haben wir?
  - Gibt der*die Nutzer*in Daten ein oder erhalten wir Input von einer anderen Quelle (z. B. über Sensoren)?
  - Hat das Programm ein Interface? Wenn ja, wie soll es aussehen?
- Welcher Output ist gewünscht?
- Welche Einschränkungen gibt es? Was kann/darf nicht passieren?
- Welche Schritte sind notwendig, um das gewünschte Ergebnis zu erhalten?

Die letzte Frage ist in der Regel nicht so leicht zu beantworten. Dabei hilft jedoch der nächste Tipp.

Zerlegt das Problem in kleinere Teilprobleme

Es ist sehr hilfreich, das anstehende Problem in kleinere Teilprobleme zu zerlegen – wie bereits aus den 4 Pfeilern des Computational Thinkings bekannt.
Sobald Teilprobleme identifiziert wurden, nehmt das kleinste oder einfachste und traut Euch an einen ersten Lösungsversuch. Ihr werdet wahrscheinlich nicht alle benötigten Schritte kennen und der erste Algorithmus wird unvollständig sein – und das ist vollkommen in Ordnung. Wenn man anfängt das erste Teilproblem zu lösen, offenbart sich oft bereits das nächste Teilproblem. Oder wenn das nächste Teilproblem bereits erkannt ist, erleichtert das Erkennen von Mustern meist die

Lösung des ersten Teilproblems. Doch anstatt einfach drauflos zu programmieren, solltet Ihr den folgenden Tipp beherzigen.

Nutzt Pseudocode

Pseudocode bedeutet, die Logik Eures Programms in Alltagssprache anstelle von Code zu schreiben. Es hilft Euch, die Schritte langsam und gründlich zu durchdenken, die das Programm durchlaufen muss, um das Problem zu lösen.
Hier ist ein einfaches Beispiel dafür, wie der Pseudocode für ein Programm aussehen könnte, das alle Zahlen bis zu einer eingegebenen Zahl ausgibt:

```
Wenn der Benutzer eine Zahl eingibt
    initialisiere eine Zählervariable und setze ihren Wert auf Null
Solange diese Variable kleiner ist als die vom Benutzer eingegebene Zahl
    erhöhe die Zählervariable um eins
Gib den Wert der Zählervariable aus
```

Nehmt Stift und Papier zur Hand und schreibt Euch den Pseudocode auf bevor Ihr anfangt, am Computer zu programmieren. Dieser Tipp ist bei komplexen Aufgaben und bei textbasierten Programmiersprachen wie Python besonders nützlich.

Erklärt Euren Code einem Quietscheentchen

Euer Code funktioniert nicht so wie erwartet? Dann erklärt Euren Code Zeile für Zeile einem Quietscheentchen (oder einer Person, die nichts vom Programmieren versteht). Das mag albern klingen, aber es ist eine bewährte Technik, die auch als »Rubber Duck Debugging« bekannt ist. Wenn Ihr Euch die Zeit nehmt, jede einzelne Zeile des Codes zu untersuchen, werdet Ihr erstaunt sein, Dinge zu erkennen, die Ihr vorher übersehen habt – insbesondere nach einer Pause.

Macht Pausen

Apropos Pausen: Macht Pausen! Es ist Zeit für eine Pause, wenn Ihr anfangt Euch müde, frustriert oder überfordert zu fühlen. Ab einem gewissen Punkt kann Beharrlichkeit mehr schaden als nützen. Außerdem fallen einem beim Programmieren oft neue Lösungen ein, wenn man eine Pause einlegt. Das passiert auch oft beim Erledigen einfacher Aufgaben wie dem Kochen einer Mahlzeit, Wäscheaufhängen oder Gassi gehen mit dem Hund.

Vielen Dank!
Wir freuen uns
über Euer Feedback.