LU03c - Rekursives Problem lösen

- Video Basics: https://www.youtube.com/watch?v=ngCos392W4w
- Towers of Hanoi: https://www.youtube.com/watch?v=rf6uf3jNjbo

Basisfall und rekursiver Fall

Jede rekursive Funktion besteht aus zwei Hauptteilen:

Basisfall: Der Basisfall ist der Teil der Funktion, der ein einfaches Problem direkt löst. Der Basisfall wird verwendet, um die Rekursion zu beenden. Ohne einen Basisfall würde die Rekursion endlos fortgesetzt.

Rekursiver Fall: Der rekursive Fall ist der Teil der Funktion, der das Problem in kleinere Teilprobleme unterteilt und sich selbst darauf aufruft.

Schritt-für-Schritt-Anleitung zur Erstellung einer rekursiven Funktion

1. **Einfachster Input (Base Case):** Das ist das Fundament einer jeden rekursiven Funktion. Bevor man versucht, die allgemeine Lösung eines Problems zu finden, muss man den einfachsten Fall oder die „Basis“ definieren, auf der die Rekursion endet. Der Base Case verhindert, dass die Rekursion unendlich wird.
2. **Herumspielen und Visualisieren:** Man experimentiert mit einigen Beispielen des Problems, um ein Verständnis dafür zu entwickeln, wie man es lösen kann. Zeichnungen oder Diagramme können dabei helfen, die Struktur des Problems zu erkennen.
3. **Schwierige Fälle mit Einfachen Vergleichen :** Hier versucht man, die komplexeren Fälle in Beziehung zu den einfacheren zu setzen. Man fragt sich, wie man ein größeres Problem in kleinere Teile zerlegen kann, die sich auf die einfacheren Fälle beziehen.
4. **Muster Generalisieren:** Nachdem man eine Beziehung zwischen den schwierigen und den einfachen Fällen gefunden hat, versucht man, eine allgemeine Lösung zu finden, die auf alle Fälle zutrifft. Das könnte eine Formel oder ein Algorithmus sein.
5. **Muster mit Base Case Kombinieren:** Schließlich kombiniert man die allgemeine Lösung mit dem Base Case, um die endgültige rekursive Funktion zu formulieren.

Beispiel: Berechnung der Fakultät

Problemstellung: Wir möchten die Fakultät einer nicht-negativen Ganzzahl n berechnen, d.h. das Produkt aller Ganzzahlen von 1 bis n.

1. Einfachster Input (Base Case)

Was ist der einfachste Fall? Die Fakultät von 0 ist 1 (definiert als $0!=1$). Rekursive Regel: Wenn $n=0$, dann ist das Ergebnis 1.

2. Herumspielen und Visualisieren

Spielen Sie mit Beispielen herum: Nehmen Sie ein paar Zahlen und berechnen Sie die Fakultät manuell, z.B. $3!=3\times2\times1\times1=6$. Visualisieren: Sie können erkennen, dass die Fakultät von n das Produkt von n und der Fakultät von $n-1$ ist.

3. Schwierige Fälle mit Einfachen Vergleichen

Da Sie beim Herumspielen gemerkt haben, dass die Fakultät von n das Produkt von n und der Fakultät von $n-1$ ist. Haben Sie gemerkt, dass die Fakultät von n als $n\times(n-1)!$ ausgedrückt werden kann.

4. Muster Generalisieren

Allgemeine Formel finden: Die rekursive Formel ist also: $n!=n\times(n-1)!$.

5. Muster mit Base Case Kombinieren

Kombinieren der Formel mit dem Base Case: Die endgültige rekursive Funktion kombiniert die allgemeine Formel mit dem Base Case.

Hier ist der Python-Code, der diese Schritte umsetzt:

```
def factorial(n):
    if n == 0: # Base Case
        return 1
    else:
        return n * factorial(n - 1) # Rekursiver Fall
```

Beispiel: Erstellung aller möglichen Kombinationen eines Strings

Problemstellung: Wir möchten alle möglichen Permutationen eines gegebenen Strings finden.

1. Einfachster Input (Base Case)

Was ist der einfachste Fall? Ein String mit einer Länge von 1 hat nur eine Permutation, nämlich sich selbst.

Rekursive Regel: Wenn die Länge des Strings gleich 1 ist, geben Sie eine Liste mit dem String selbst zurück.

2. Herumspielen und Visualisieren

Spielen Sie mit Beispielen herum: Nehmen Sie einen String wie AB und finden Sie alle möglichen Kombinationen (AB, BA).

Visualisieren: Sie können erkennen, dass die Permutation eines Strings die Verbindung eines jeden Zeichens mit der Permutation des Restes des Strings ist.

3. Schwierige Fälle mit Einfachen Vergleichen

Relation zwischen schwierigen und einfachen Fällen: Die Permutationen eines Strings können gefunden werden, indem jedes Zeichen im String als Startzeichen genommen und mit den Permutationen des restlichen Strings verbunden wird.

4. Muster Generalisieren

Allgemeine Formel finden: Die rekursive Formel besteht darin, jedes Zeichen des Strings als Startpunkt zu nehmen und mit den Permutationen des Rests des Strings zu kombinieren.

5. Muster mit Base Case Kombinieren

Kombinieren der Formel mit dem Base Case: Die endgültige rekursive Funktion kombiniert die allgemeine Formel mit dem Base Case. Hier ist der Python-Code, der diese Schritte umsetzt:

```
def permutations(s):
    if len(s) == 1: # Base Case
        return [s]
    perm_list = [] # Liste, um die Permutationen zu speichern
    for char in s:
        remaining_chars = s.replace(char, '', 1)
        remaining_permutations = permutations(remaining_chars) # Rekursiver
Aufruf
        for perm in remaining_permutations:
            perm_list.append(char + perm)
    return perm_list
```

M323-LU03, M323-C1B, M323-C1I

© Kevin Maurizi

From:
https://wiki.bzz.ch/ - **BZZ - Modulwiki**

Permanent link:
https://wiki.bzz.ch/modul/m323/learningunits/lu03/rekursion2

Last update: **2026/08/18 10:18**