LU03.A01 - Rekursive Suche in einem Verzeichnisbaum

Stellen Sie sich einen Verzeichnisbaum auf einem Computer vor, in dem jedes Verzeichnis Unterverzeichnisse oder Dateien enthalten kann. Schreiben Sie eine rekursive Funktion, die eine Datei in diesem Verzeichnisbaum sucht.

Sie können den Verzeichnisbaum als verschachtelte Dictionary-Struktur darstellen:

```
directory_tree = {
    'name': 'root',
    'path': '/'
    'type': 'directory',
    'children': [
        {
            'name': 'subdir1',
            'path': '/subdir1'
            'type': 'directory',
            'children': [
                {
                    'name': 'file2.txt',
                    'path': '/subdir1/file2.txt''
                    'type': 'file'
                }
                {...}
            ]
        },
        {
            'name': 'file.txt',
            'path': '/file.txt''
            'type': 'file'
        }
    ]
}
```

Die Funktion sollte den Pfad zur gesuchten Datei zurückgeben oder `None`, wenn die Datei nicht gefunden wird.

Vorlage

```
directory_tree = {
    'type': 'directory',
    'name': 'root',
```

```
    'path': '/',
    'children': [
        {
            'type': 'directory',
            'name': 'home',
            'path': '/home',
            'children': [
                {
                    'type': 'directory',
                    'name': 'user',
                    'path': '/home/user',
                    'children': [
                        {'type': 'file', 'name': 'file1.txt', 'path':
'/home/user/file1.txt'},
                        {'type': 'file', 'name': 'file2.txt', 'path':
'/home/user/file2.txt'},
                    ],
                },
                {'type': 'file', 'name': 'readme.md', 'path':
'/home/readme.md'},
            ],
        },
        {
            'type': 'directory',
            'name': 'etc',
            'path': '/etc',
            'children': [
                {'type': 'file', 'name': 'config.yaml', 'path':
'/etc/config.yaml'},
                {
                    'type': 'directory',
                    'name': 'nginx',
                    'path': '/etc/nginx',
                    'children': [
                        {'type': 'file', 'name': 'nginx.conf', 'path':
'/etc/nginx/nginx.conf'},
                        {
                            'type': 'directory',
                            'name': 'sites-enabled',
                            'path': '/etc/nginx/sites-enabled',
                            'children': [
                                {'type': 'file', 'name': 'default', 'path':
'/etc/nginx/sites-enabled/default'}
                            ],
                        },
                    ],
                },
            ],
        },
```

```
    ],
}

def find_file(name, directory):
    # Your code goes here

if __name__ == '__main__':
    path = find_file('config.yaml', directory_tree)
    print(path)  # Sollte den Pfad zur Datei ausgeben
```

Vorgehen

Gehen Sie nach der 5-Schritte Methode vor, überlegen Sie sich für jeden Schritt die Lösung und tragen Sie alles als funktionierendes Programm zusammen:

1. **Einfachster Input (Base Case)**
2. **Herumspielen und Visualisieren**
3. **Schwierige Fälle mit Einfachen Vergleichen**
4. **Muster Generalisieren**
5. **Muster mit Base Case Kombinieren**

Hinweise

Einfachster Input (Base Case): Der einfachste Fall tritt auf, wenn ein Verzeichnis keine Unterordner (Kinder) hat oder wenn der gesuchte Dateiname im aktuellen Verzeichnis gefunden wird. In diesen Fällen kann die Rekursion beendet werden.

```
FUNKTION finde_element(name, element):
    // Base Case: Wenn das Element im aktuellen Element gefunden wird
    WENN element['type'] == 'file' und element['name'] == name DANN
        RÜCKGABE Element['path']
    ENDE_WENN
    ...
```

Herumspielen und Visualisieren: Um das Problem besser zu verstehen, könnte man mit einem kleinen Verzeichnisbaum experimentieren und versuchen, den Weg zu einer bestimmten Datei manuell zu finden. Diagramme oder Skizzen des Baums könnten dabei hilfreich sein.

Schwierige Fälle mit Einfachen Vergleichen: In einem komplexen Verzeichnisbaum ist es hilfreich, die Suche in jedem Unterordner als ein separates, einfacheres Problem zu betrachten. Die Suche in einem Unterordner ist identisch wie die Suche im übergeordneten Ordner.

Muster Generalisieren: Nachdem man die Beziehung zwischen den komplexen und den einfachen Fällen gefunden hat, könnte man eine allgemeine Lösung entwickeln. In diesem Fall wäre das eine rekursive Funktion, die durch jeden Unterordner geht und die gleiche Suche darin ausführt, bis der gesuchte Dateiname gefunden wird oder kein Unterordner mehr übrig ist.

```
FUNKTION finde_element(name, element):
```

```
    ...
    // Wenn das Element Kinder hat, durchlaufe sie
    WENN element['type'] == 'directory' und 'children' IN Element DANN
        FÜR JEDES kind IN element['children']
            pfad = find_element(name, kind)  // Rekursiver Aufruf
            WENN pfad EXISTIERT DANN // Wenn der Pfad gefunden wurde, gebe
ihn zurück
                RÜCKGABE pfad
            ENDE_WENN
        ENDE_FÜR
    ENDE_WENN
    ...
```

Muster mit Base Case Kombinieren: Die endgültige rekursive Funktion würde die allgemeine Lösung (durch jeden Unterordner gehen) mit dem Base Case (keine Unterordner oder Dateiname gefunden) kombinieren, um eine vollständige Lösung für das Problem zu bieten.

```
 // Base Case: Wenn das Element im aktuellen Element gefunden wird
    WENN element['type'] == 'file' und element['name'] == name DANN
        RÜCKGABE Element['path']
    ENDE_WENN
    // Wenn das Element Kinder hat, durchlaufe sie
    WENN element['type'] == 'directory' und 'children' IN Element DANN
        FÜR JEDES kind IN element['children']
            pfad = find_element(name, kind)  // Rekursiver Aufruf
            WENN pfad EXISTIERT DANN // Wenn der Pfad gefunden wurde, gebe
ihn zurück
                RÜCKGABE pfad
            ENDE_WENN
        ENDE_FÜR
    ENDE_WENN
    // Wenn das Element nicht gefunden wird
    RÜCKGABE None
```

© Kevin Maurizi

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

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

Last update: **2024/03/28 14:07**