Python (Repetition) Fragen & Antworten

Frage 6.1

Was bedeutet es, wenn mit `L[i][j]` oder `L[i][j][k]` auf eine Liste `L` zugegriffen wird?

Frage 6.2

Welches sind die Regeln zur Bildung von Slices (Teillisten)?

Frage 6.3

Wann ist der Wert eines Python-Ausdrucks eine Liste?

Frage 6.4

Worin besteht der Unterschied zwischen `L.pop()` und `L.pop(i)`?

Frage 6.5

Worin besteht der Unterschied zwischen den Listen-Methoden `L.pop(3)` und `L.remove(3)`?

Frage 6.6

Wann teilen sich zwei Variablen eine Liste?

Frage 6.7

Was ist in Python eine *List-Comprehension*?

Frage 7.1

Welches sind die drei Hauptgründe für die Verwendung von Funktionen?

Frage 7.2

Zu welchem Zeitpunkt wird in Python eine Funktion erzeugt?

Frage 7.3

Wann wird der Code ausgeführt, der innerhalb einer Funktionsdefinition steht?

Frage 7.4

Welche Möglichkeiten gibt es, beim Aufruf eine Funktion die aktuellen Parameter den formalen Parametern zuzuweisen?

Frage 7.5

Wie kann eine Funktion ein von ihr berechnetes Resultat an die aufrufende Instanz übermitteln?

Frage 7.6

Was gibt eine Funktion zurück, die keine `return`-Anweisung in ihrer Definition enthält?

Frage 7.7

Was bedeutet es, wenn bei der Definition einer Funktion die Parameter mit Standardwerten belegt weden?

Frage 7.8

Wie lauten die beiden wichtigsten Regeln für die Gültigkeit von Namen innerhalb und ausserhalb von Funktionen?

Frage 7.9

Wie verfährt der Python-Interpreter mit Variablen, die innerhalb einer Funktion stehen?

Frage 7.10

Wie berechnet Python den Wert `f(6)` für die folgende rekursiv definierte Funktion?

```
def f(x):
    if x < 3
        return 5
    else:
        return x * f(x - 2)
```

Frage 7.11

Kann in Python eine Funktion Parameter (Argument) einer anderen Funktion sein?