karriere tutor®
Gemeinsam. Digital. Erfolgreich.

C++

Zeiger und Arrays

Aufgabe 0: Aussagen einordnen

Wahr oder falsch? Begründe deine Antwort.

a) Ein Zeiger speichert die Adresse einer Variablen.

b) Ein `const`-Zeiger kann seinen Wert ändern, aber nicht die Adresse, auf die er zeigt.

c) Ein `constexpr` muss während der Kompilierung bekannt sein.

d) Ein Array kann nach der Initialisierung seine Größe dynamisch ändern.

e) Ein C-String ist ein Array von `char`, das mit dem Nullterminator (`\0`) endet.

f) `const` schützt eine Variable nur vor der Modifikation durch den Programmierer, nicht jedoch vor einer Modifikation durch Zeiger.

g) Ein Zeiger auf `const` erlaubt keine Änderung der Daten, auf die er zeigt.

h) `constexpr`-Variablen können während der Laufzeit geändert werden.

i) Zeigerarithmetik kann verwendet werden, um durch die Elemente eines Arrays zu iterieren.

j) C-Strings werden immer dynamisch allokiert.

Aufgabe 1: Array-Manipulation mit Zeigern

Schreibe eine Funktion, die ein Array von Ganzzahlen als Zeiger empfängt, den größten Wert im Array findet und dessen Adresse zurückgibt. Implementiere zusätzlich eine `main`-Funktion, die das Programm testet.

Bsp.: `int* findeGroesstenWert(int* arr, int groesse)`

Aufgabe 2: Vergleich von `const` und `constexpr`

Schreibe ein Programm, das zeigt, wann es sinnvoll ist, `const` und wann `constexpr` zu verwenden. Erkläre durch Kommentare, warum du dich in jedem Fall für das eine oder das andere entschieden hast.

Aufgabe 3: C-Strings und Zeiger

Implementiere eine Funktion, die einen C-String als Eingabe erhält und den String umkehrt, ohne eine zusätzliche Kopie zu erstellen. Verwende dabei Zeigerarithmetik, um die Zeichen zu tauschen.

Aufgabe 4: Mehrdimensionale Arrays mit Zeigern

Schreibe ein Programm, das ein zweidimensionales Array (Matrix) von `int` mit einer festen Größe (z. B. 3x3) erstellt. Implementiere eine Funktion, die die Matrix transponiert (Zeilen werden zu Spalten) und verwende dafür Zeigerarithmetik.

Bsp.: `void transponiereMatrix(int matrix[SIZE][SIZE])`

Aufgabe 5: Konstante Zeiger und Speicherverwaltung

Schreibe ein Programm, das dynamisch Speicher für ein Array von `const`-Zeigern auf `int` alloziert. Zeige, wie der Speicher am Ende korrekt freigegeben wird, und erkläre den Unterschied zwischen `const int*` und `int* const` in den Kommentaren.

Bsp. Um dynamsich Speicher für die Werte zu allokieren: `constArr[0] = new int(10);`

Der Speichersollte dann folgendermaßen wieder freigegeben werden:

```
for (int i = 0; i < SIZE; i++) {
    delete constArr[i];
}
```

Aufgabe 6: Zufällige Tabellenkalkulation

Schreibe eine kleine 'Tabellenkalkulation'. Für die Tabelle ist ein Feld mit 10x10 Einträgen anzulegen. Fülle dieses Feld mit Zufallszahlen im Bereich 0...9. Gebe die Tabelle sowie die Zeilen- und Spaltensummen aus. Achte bei der Ausgabe auf eine saubere Formatierung. Auf der nächsten Seite findet ihr ein Beispiel für eine Ausgabe.

Für die Formatierung bindet die Bib `<iomanip>` ein.
Für den Zufallszahlengenerator bindet die Bib `< cstdlib>` ein.
Für die Initialisierung des Zufallszahlengenerators bindet die Bib `< ctime>` ein.

```
// Zufallszahlengenerator initialisieren
std::srand(std::time(0));

// Zufallszahl zwischen 0 und 9
std::rand() % 10;
```

```
// Formatierte Ausgabe der Tabelle
std::setw(2)

// Trennlinie zwischen Tabelle und Spaltensummen
std::string(cols * 3, '-')
```

Ausgabe:

```
  1   7   4   0   9   4   8   8   2   4 :   47
  5   5   1   7   1   1   5   2   7   6 :   40
  1   4   2   3   2   2   1   6   8   5 :   34
  7   6   1   8   9   2   7   9   5   4 :   58
  3   1   2   3   3   4   1   1   3   8 :   29
  7   4   2   7   7   9   3   1   9   8 :   57
  6   5   0   2   8   6   0   2   4   8 :   41
  6   5   0   9   0   0   6   1   3   8 :   38
  9   3   4   4   6   0   6   6   1   8 :   47
  4   9   6   3   7   8   8   2   9   1 :   57
-----------------------------------------------
 49  49  22  46  52  36  45  38  51  60
```

Falls ihr Fragen habt oder Probleme auftreten, meldet euch am besten sofort. Ich wünsche euch viel Spaß bei der Implementierung.