karriere tutor®
Gemeinsam. Digital. Erfolgreich.

C++

Grundlagen der Programmierung

Aufgabe 0: Aussagen einordnen

Wahr oder falsch? Begründe deine Antwort.

a) Der Datentyp `int` wird verwendet, um Ganzzahlen zu speichern.

b) Die `main`-Funktion ist optional und muss nicht in jedem C++-Programm vorhanden sein.

c) Ein `double`-Wert kann Dezimalstellen enthalten.

d) Variablen müssen deklariert werden, bevor sie verwendet werden können.

e) Konstanten können ihren Wert während der Programmausführung ändern.

f) Literale sind feste Werte im Code, wie z.B. Zahlen oder Zeichenketten.

g) Ein `char` speichert genau einen Buchstaben oder ein Zeichen.

h) Eine Variable in C++ darf nicht mit einer Zahl beginnen.

i) Der Operator = wird in C++ für den Vergleich von zwei Werten verwendet.

j) Ein String-Literal wird in doppelte Anführungszeichen " eingeschlossen.

Aufgabe 1: Datentypen und Variablen

Deklariere die folgenden Variablen und weisen Sie ihnen jeweils einen Wert zu:

- Eine Ganzzahl-Variable `age`.
- Eine Gleitkommazahl-Variable `height`.
- Eine Zeichen-Variable `initial`.
- Eine Zeichenketten-Variable `name`.
- Eine boolesche Variable `isStudent`.

Schreibe ein kurzes Programm, das diese Variablen deklariert, ihnen Werte zuweist und dann diese Werte auf der Konsole ausgibt.

Aufgabe 2: Die `main`-Funktion

Erstelle ein einfaches C++-Programm, das folgende Anforderungen erfüllt:

- Die `main`-Funktion soll die Steuerung des Programms übernehmen.
- Innerhalb der main-Funktion sollen zwei Ganzzahlen deklariert und initialisiert werden.
- Das Programm soll die Summe dieser beiden Zahlen berechnen und das Ergebnis auf der Konsole ausgeben.
- Kommentiere den Code ausführlich, um zu erklären, was jede Zeile macht.

Aufgabe 3: Verwendung von Konstanten

Schreibe ein Programm, das die Fläche eines Kreises berechnet. Verwende dabei eine Konstante für den Wert von π (`Pi = 3.14159`).

- Deklariere eine Konstante `PI`.
- Deklariere eine Variable `radius` für den Radius des Kreises. Radius vom Benutzer eingeben.
- Berechne die Fläche des Kreises mit der Formel `Fläche = PI * radius * radius`.
- Gebe den Radius und die berechnete Fläche auf der Konsole aus.

Aufgabe 4: Arbeiten mit Literalen

Erstelle ein Programm, das verschiedene Arten von Literalen verwendet und diese ausgibt:

- Ein Integer-Literal, das eine Dezimalzahl darstellt.
- Ein Integer-Literal, das eine Hexadezimalzahl darstellt. 0x
- Ein Floating-Point Literal, das eine Gleitkommazahl darstellt.
- Ein Zeichen-Literal.
- Ein String-Literal.

Das Programm soll alle diese Literale auf der Konsole ausgeben und kommentieren, um zu zeigen, welche Art von Literal verwendet wurde.

Aufgabe 5: Konversionsberechnungen und Datentypen

Schreibe ein Programm, das mehrere Einheitenkonversionen basierend auf Benutzereingaben durchführt. Das Programm soll folgende Schritte ausführen:

1. Eingabe des Benutzers:
   - Der Benutzer gibt eine Temperatur in Grad Celsius ein, die das Programm in Fahrenheit und Kelvin umrechnen soll.
   - Der Benutzer gibt eine Länge in Metern ein, die das Programm in Zentimeter, Millimeter und Kilometer umrechnen soll.
   - Der Benutzer gibt ein Gewicht in Kilogramm ein, das das Programm in Gramm und Tonnen umrechnen soll.

2. Konstanten und Berechnungen:
   - Verwende Konstanten, um die Umrechnungsfaktoren für die Einheiten festzulegen.
   - Berechne die Umrechnungen basierend auf den Benutzereingaben.

Konstanten werden groß geschrieben und mit SNAKE_CASE verbunden

3. Fehlerbehandlung:
   - Das Programm soll überprüfen, ob die Eingaben gültig sind (z.B. keine negativen Temperaturen im Bereich unter dem absoluten Nullpunkt, keine negativen Längen oder Gewichte).
   - Geben Sie eine Fehlermeldung aus und fordern Sie den Benutzer zur erneuten Eingabe auf, falls die Eingabe ungültig ist.

4. Ausgabe:
   - Das Programm gibt alle berechneten Werte aus und formatiert die Ausgabe klar und lesbar.

Hinweis:

Stelle sicher, dass du für die Berechnungen die korrekten Datentypen verwendst, um Präzisionsverluste zu vermeiden.

Aufgabe 5: Simples Finanzmodell

Erstelle ein Programm, das eine einfache Simulation für die Verwaltung von Kontoständen eines Benutzers durchführt. Das Programm soll die folgenden Funktionen und Anforderungen erfüllen:

1. Initialisierung:
   - Deklariere eine Konstante für den Anfangskontostand (`INITIAL_BALANCE`), z.B. 1000.0 EUR.
   - Erstelle Variablen für den aktuellen Kontostand, die Anzahl der Transaktionen und den Gesamtsaldo.
2. Konstanten und Berechnungen:
   - Zeige ein Menü mit folgenden Optionen an:
     - Einzahlung vornehmen
     - Abhebung vornehmen
     - Kontostand anzeigen
     - Programm beenden
   - Verwende eine Schleife, um das Menü dem Benutzer so lange anzuzeigen, bis dieser das Programm beendet.

3. Transaktionslogik:
    - Bei einer Einzahlung erhöht sich der Kontostand um den eingegebenen Betrag.
    - Bei einer Abhebung verringert sich der Kontostand um den eingegebenen Betrag. Das Programm soll überprüfen, ob der Abhebungsbetrag größer als der aktuelle Kontostand ist und in diesem Fall eine Fehlermeldung ausgeben.
    - Die Anzahl der Transaktionen (Einzahlungen und Abhebungen) soll gezählt und angezeigt werden.

4. Berichterstattung:
    - Bei der Auswahl von "Kontostand anzeigen" gibt das Programm den aktuellen Kontostand und die Anzahl der Transaktionen aus.
    - Wenn der Benutzer das Programm beendet, soll eine Zusammenfassung mit dem Startsaldo, dem Endsaldo und der Gesamtzahl der Transaktionen angezeigt werden.

Hinweis:

Achte auf eine sinnvolle Verwendung von Datentypen und Konstanten, um Präzision und Stabilität des Programms zu gewährleisten. Verwende geeignete Schleifen- und Kontrollstrukturen, um die Benutzerinteraktion effizient zu gestalten.

Diese Aufgaben fordern dich heraus, die Grundlagen der C++-Programmierung in komplexeren Szenarien anzuwenden. Sie erfordern ein gutes Verständnis von Variablen, Datentypen, Konstanten, Kontrollstrukturen und der `main`-Funktion. Arbeite sorgfältig und teste deine Programme gründlich, um sicherzustellen, dass sie korrekt und effizient funktionieren. Viel Erfolg!