Projekt Kopfrechentrainer

Der zu entwickelnde Rechentrainer soll folgende Eigenschaften besitzen:

- Der Rechner erzeugt selbstständig zwei Zufallszahlen im Bereich von 1 bis 20 und erwartet als Eingabe das Produkt beider Zahlen.
- Bei korrekter Eingabe erscheint im Ergebnisfeld „Richtig!“ bzw. „Falsch!“.

0. Neues Projekt anlegen !!!

1. Erstellen des Formulars:

- Erstellen Sie eine Form nach nebenstehender Vorlage.
- Wählen Sie die **Bezeichner** der Komponenten (**Eigenschaft Name** im Objektinspektor) sinnvoll.
- Verwenden Sie für die Überschrift und das Ergebnisfeld die **Komponente Panel** im **Register Standard**.
- Setzen Sie die Eigenschaft **ReadOnly** der ersten beiden Editfelder auf **True**, um Eingaben des Benutzers in diese Felder zu vermeiden.
- Die drei Knöpfe **„Neu“, „Test“** und **„Abbruch“** sind **BitButton-Komponenten** (Register Additional) mit den Eigenschaften **„bkRetry“, „bkOK“** und **„bkAbort“.**

2. Implementierung

Bemerkung: Testen Sie Ihr Programm möglichst oft (d.h. nach jeder Änderung) durch einen Neustart, um mögliche Quelltext- bzw. Laufzeitfehler frühzeitig und damit eindeutig aufzudecken.

- Sie benötigen zwei Zufallszahlen im Bereich zwischen 1 und 20.
- Deklarieren sie diese beiden Zufallszahlen als **globale Variablen** (am Anfang des Implementation-Teils).

1. Implementieren Sie die **Prozedur** `BitBtnRetryClick` (Doppelklick auf den entspechenden BitButton), welche
   - die beiden Zufallszahlen erzeugt (`Random(20)+1`) und
   - diese in den ersten beiden Editfeldern ausgibt (`IntToStr(Zufallszahl1)`) und
   - den Eingabefokus auf das dritte Editfeld setzt (`EdEingabe.SetFocus`).

   - Um zu garantieren, daß bei jedem Programmstart nicht die selbe Folge von Zufallszahlen erscheint, soll bei Erzeugen der Form die Prozedur Randomize aufgerufen werden (Doppelklick auf Ereignis **OnCreate** der Komponente **FoRechentrainer** im Objektinspektor; siehe letztes Arbeitsblatt 05).

   - Um schon zum Programmstart die in der Prozedur `BitBtnRetryClick` implementierte Funktionalität bereitzustellen, rufen Sie diese beim Aktivieren der Form auf (Auswahlfeld des Ereignisses **OnActivate** der Komponente **FoRechentrainer** im Objektinspektor)

2. Implementieren Sie die **Prozedur** `BitOKClick` (Doppelklick auf den entspechenden BitButton), welche
   - das Produkt beider Zufallszahlen
     mit der momentanen Eingabe `EdEingabe.Text` vergleicht, und

   - je nach Ergebnis (`if...then...else...`) dieses Vergleichs

     - eine **Meldung** im Ergebnispanel ausgibt:
       - „Das war richtig!“ in grüner Schrift bzw.
       - „Das war falsch!“ in roter Schrift.

     - und den **Eingabefokus**
       - auf das Editfeld (bei falscher Eingabe) bzw.
       - auf den BitBtnRetry (Neues Produkt) setzt (z.B. `BitBtnRetry.SetFocus`).

3. Zusatzaufgaben:

1. Strukturieren Sie Ihren Quelltext, falls noch nicht geschehen, indem Sie zusammengehörende bzw. gleichwertige Teile gleich weit einrücken. Meist erfolgt die Einrückung um **zwei Leerzeichen**. Durch die Tastenkombination **Strg+Shift+I** (ein Zeichen nach rechts) bzw. **Strg+Shift+U** (nach links) können Sie mehrere Zeilen gleichzeitig ein- bzw- ausrücken. Folgende Strukturierung hilft bei späterer Fehlersuche:

- Zusammengehörende Schlüsselwörter **`begin`** und **`end`** werden immer gleich weit eingerückt. Damit lassen sich fehlende oder überzählige Schlüsselwörter leichter erkennen.
- Der Quelltext zwischen **`begin`** und **`end`** wird um zwei Zeichen eingerückt.
- Bei einer **`if...then...else..`** Anweisung werden die Schlüsselwörter **`then`** und **`else`** gleich weit, aber um zwei Zeichen relativ zur if–Bedingung eingerückt.
- Ein durch **`begin`** und **`end`** geklammerte Verbundanweisung wird ebenfalls eingerückt.

2. Statt der Meldung „Das war falsch!“ soll nun die Aufschrift „1. Fehler“ usw. erscheinen, wobei Sie für den Zähler eine weitere globale Variable einführen müssen.

3. Nach jedem 5. Fehler soll die Meldung „Mensch, lern endlich!“ erscheinen.
Verwenden Sie hierzu in Ihrer Bedingung die arithmetische Operation **`mod`**, welche den Rest bei der Ganzzahldivision zurückgibt. Informieren Sie sich über die arithmetischen Operationen **`mod`** und **`div`**, indem das Schlüsselwort **`mod`** markieren und die Funktionstaste **F1** drücken.

4. Nun sollen nicht nur die beiden Operatoren zufällig bestimmt werden, sondern auch die Operation. Überarbeiten Sie Ihr Programm so, daß in der **Prozedur** **`BitBtnRetryClick`** zusätzlich zu den oben genannten Funktionalitäten

- zufällig eine der vier Grundrechenarten ausgewählt wird (**`random(4)`**)
- die Beschriftung des Labels geändert wird (**`LaZeichen.Caption := '+';`**)
- das jeweilige Ergebnis berechnet wird und als globale Variable zur Verfügung steht.

Beachten Sie, daß dadurch die beiden Zufallszahlen keine globalen Variablen mehr sein müssen.
Achten Sie immer darauf, so wenig wie möglich globale Variablen zu verwenden!
Überprüfen Sie Ihr Programm dementsprechend.

Weitere Aufgaben mit Entscheidungen:

1. Lösung lineare und quadratischer Gleichungen:
Implementieren Sie die Aufgaben der letzten Woche zur Lösung linearer und quadratischer Gleichungen.

2. Schaltjahr:
Nach Eingabe einer Jahreszahl soll bestimmt werden, ob es sich um ein Schaltjahr handelt oder nicht.
(Tipp: Informieren Sie sich über die arithmetischen Operatoren **`div`** und **`mod`** in der Delphi-Hilfe !)

3. Body-Mass-Index (BMI):
Die moderne Ernährungswissenschaft rechnet mit dem Body-Mass-Index und

„dividiert die Masse in kg durch das Quadrat der Körpergröße in m.“

Ihr Programm soll den BMI nach Eingabe von Geschlecht, Körpergröße und Masse berechnen und die entsprechende Gewichtsklasse ausgeben:

BMI	männlich	weiblich	Gewichtsklasse
unter	20	18	Untergewicht
unter	25	24	Normalgewicht
unter	30	30	Übergewicht
über	30	30	medizinisch bedenklich