Übung 8 "Modularisierung"

Theoriefrage:

Welche Vorteile sind mit der Modularisierung verbunden?

Was bedeutet CALL-BY-REFERENZ?

Was bedeutet eine Variable, die Global definiert ist?

Welche Bedeutung haben globale Datentypen für Funktionsbausteine?

Was ist der Unterschied von Funktionsgruppe und Funktionsbaustein?

Was bedeutet IMPORTING/EXPORTING jeweils aus Sicht des ABAP-Programms und des Funkionsbausteins?

Warum müssen Gobal definierte Datentypen zur Typisierung einer Funktionsbausteinschnittstelle verwendet werden?

Warum ist die Zuweisung von Variablen beim Import und Export unterschiedlich. Was ist hier auffällig?

ABAP-Programmierung

Aufgabe 1: Taschenrechner mit Unterprogramm (ZSemester_###_UEB8_A1)

Den Taschenrechner haben Sie bereits unter dem Programmnamen ZSemester_###_UEB6_A4) erstellt. Das Programm erfüllt die grundlegenden mathematischen Funktionen eines einfachen Taschenrechners und beherrscht die Funktionen *Addition, Subtraktion, Multiplikation, Division* – siehe Übung 06 - Taschenrechner.

Jetzt sollen die Rechenoperationen in vier Unterprogramme mit dem Namen FORM ADD, FORM SUB, FORM MULT und FORM DIV gekapselt werden.

Erweitern Sie das Programm darüber hinaus mit einer % Funktion, die ebenfalls in einem Unterprogramm programmiert wird. Erweitern Sie hierzu die Bildschirmeingabe ihres Ursprungprogramms um die Eingabemöglichkeit von % in der Operatorauswahl.

Bildschirmaufbau mit PARAMETERS:
Operatorauswahl +,-,*,/, % __
Zahl 1: _____
Zahl 2: _____

***Info**: Bei Prozent ist die Zahl2 der Prozentsatz, die Zahl1 der Wert, von dem die Prozente gerechnet werden sollen (erg = zahl1 /100 * zahl2).

Lösungshinweis:

Definieren Sie für die Unterfunktionen jeweils USING-Parameter (call-by-value) zur Übergabe der Operanden (pv_zahl1, pv_zahl2). Definieren Sie einen CHANGING-Parameter (call-by-referenz) zur Rückgabe des Ergebnisses (cv_ergebnis).

Geben Sie bei Division durch 0 einen Hinweis aus.

Erweitern Sie die CASE-Anweisung im Hauptprogramm um einen Zweig für ‚%'. Rufen Sie die Berechnungen über ein Unterprogramm auf.

Lassen Sie sich den Aufruf des Unterprogramms generieren, indem Sie die Drag&Drop-Funktionalität des Navigationsbereichs verwenden.

Debugging

Schauen Sie sich den Programmablauf an. Nutzen Sie die Buttons um Unterprogramme zu überspringen oder zu verlassen.

Aufgabe 2: Funktionsbaustein zur Berechnung einer Dauer (ZSemester_###_UEB8_A2)

Legen Sie das Programm ZSemester_###_UEB8_A2 an. Definieren Sie ein Datenobjekt für die Dauer (gv_duration, TYPE p LENGTH 16). Definieren Sie die mit Parameters die Eingabeparameter für die Variablen:

- Startdatum (pa-sdate) und Startzeit (pa-stime)
- Enddatum (pa-edate) und Endzeit (pa-edate)

Zur Typisierung verwenden Sie bitte die Standardtypen d und i.

Binden Sie nun den fertigen Funktionsbaustein ‚DURATION_DETERMINE' ein. Suchen Sie den Funktionsbaustein ‚DURATION_DETERMINE' und binden diesen

mit dem Muster in ihr Programm ein. Kennzeichen für Kommentarzeilen bei den notwendigen Parameters entfernen und die Programmvariablen (pa-sdate, pa-stime, pa-edate, pa-edate und gv_duration) zuordnen.

IF-Anweisung erweitern und bei erfolgreicher Ausführung (Returncode

SY-SUBRC = 0) die Dauer mit einem beschreibenden Text per WRITE ausgeben.

Bei nicht erfolgreicher Ausführung eine einfache Fehlermeldung per WRITE ausgeben.

Beispielcode zur Orientierung.

```
DATA: gv_duration TYPE p LENGTH 16.

CALL FUNCTION 'DURATION_DETERMINE'
EXPORTING
UNIT = 'MIN'
IMPORTING
DURATION = GV_DURATION
CHANGING
START_DATE = PA_SDATE
START_TIME = PA_STIME
END_DATE = PA_EDATE
END_TIME = PA_ETIME
EXCEPTIONS
FACTORY_CALENDAR_NOT_FOUND = 1
DATE_OUT_OF_CALENDAR_RANGE = 2
DATE_NOT_VALID = 3
UNIT_CONVERSION_ERROR = 4
SI_UNIT_MISSING = 5
PARAMETERS_NOT_VALID =6
OTHERS = 7.
```

Alternative für die Einbindung mit Muster:

Button ‚Anderes Objekt', Reiter Funktionsgruppe, ……Funktionsbaustein mit Button ‚Objektliste anzeigen' in die Objektliste übernehmen. Funktionsbaustein mit dem Mauszeiger an die betreffende Stelle im Programm ziehen.

Debugging das Programm.

Schauen Sie sich den Programmablauf an. Nutzen Sie die Button um Funktionsbausteine zu überspringen oder zu verlassen.

Aufgabe 3: Taschenrechner mit Funktionsbaustein (ZSemester_###_UEB8_A3)

Realisieren Sie den Taschenrechner mit Hilfe eines Funktionsbausteines, der die Rechenoperationen *Addition, Subtraktion, Multiplikation, Division* bereitstellt.

Gehen Sie wie folgt vor:

Schritt 1: Anlage der Funktionsgruppe

Legen Sie eine **Funktionsgruppe** mit dem Namen **ZSemester_###_fg** an. Zur Anlage gibt es mehrere Möglichkeiten:

Alternative 1

Paket auswählen zu dem die Funktionsgruppe angelegt werden soll. Über das Kontextmenü zum Paket Anlegen – Funktionsgruppe – Namen eingeben.

Alternative 2

Button „Objekt bearbeiten“ – Reiter Funktionsgruppe –Gruppennamen eingeben – Button „Anlegen“

Kurztext: Bezeichnung der Gruppe (Funktionsgruppenname: ZSemester_###_FG)

Die Funktionsgruppe wurde angelegt und besteht am Anfang aus einem **Functionpool** (Rahmenprogramm) und 2 **Include-Programmen** (1 für die Datendeklaration, 1 für die Liste der Funktionsbausteine).

Schritt 2: Anlage von Globalen Datentypen im DD für die Schnittstelle im Funktionsbaustein

Für die Schnittstellen im Funktionsbaustein benötigen wir im DD globale Datenelemente. Legen Sie diese Datenelemente mit den nachfolgenden Namen an.

ZSemester_XXX_RECH (Zuordnung Domäne **ZSemester_XXX_RECH,** Bezeichnung Operator)

ZSemester_XXX_EIN1 und **ZSemester_XXX_EIN2** (Type dec(12,2), Bezeichnung Zahl 1, Zahl 2)

ZSemester_XXX_ERGEBNIS (Type Dec 12, 2 Nachkommastellen, Bezeichnung Ergebnis)

Schritt 3: Funktionsbaustein erstellen

Legen Sie nun den Funktionsbaustein **ZSemester_###_trech** an. Auch hier gibt es zwei Wege aus dem SE80 anzulegen.

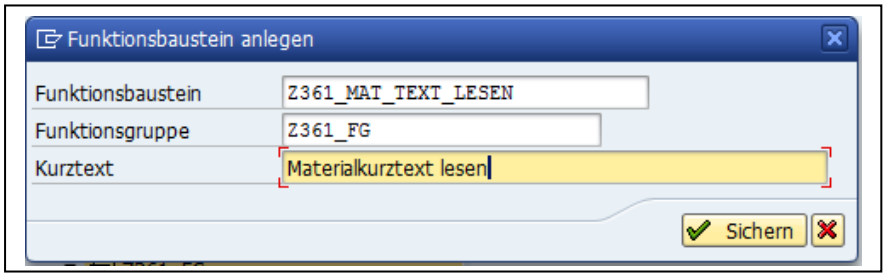
Alternative 1

Funktionsgruppe auswählen zu dem der Funktionsbaustein angelegt werden soll. Über das Kontextmenü zur Funktionsgruppe Anlegen – Funktionsbaustein .

Alternative 2

Button „Objekt bearbeiten“ – Reiter Funktionsgruppe –Funktionsbaustein auswählen – Namen eingeben - Button „Anlegen“

Namen, Funktionsgruppe und Bezeichnung (Kurztext) eingeben (Name Funktionsbaustein: **ZSemester_###_trech** Name Funktionsgruppe **ZSemester_###_fg**.



Sichern Sie die Eingaben und Programmieren Sie die Schnittstellen (Import, Export) und den Quelltext für die Taschenrechnerfunktion. Für die Typisierung der Schnittstelle verwenden Sie die in Schritt 2 angelegten globalen Datentypen. Als Quelltext legen Sie einen CASE-Konstrukt an (siehe vorherige Übungen).

Schritt 4: Erstellen Sie ein ABAP-Programm mit Aufruf des Funktionsbausteins ZSemester_###_trech

Erstellen Sie das Programm ZSemester_###_UEB8_A3 und nutzen Sie den Funktionsbaustein zur Berechnung der Ergebnisse. Kopieren Sie sich hierzu das Programm **ZSemester_###_UEB6_A1 (Taschenrechner) als Vorlage.**

Parameter wie bisher, allerdings **typisiert** auf die neu angelegten **globalen Datentypen**. Der Parameter sollte um den Zusatz ‚**VALUE CHECK**‘ ergänzt werden.

Definieren Sie das Datenobjekt **GD_ERGEBNIS** typisiert mit dem **globalen Datentyp**.

Division durch Null verhindern. (Die Gültigkeit der Operatoren muss nicht mehr abgefragt werden).

Funktionsbaustein aufrufen und dass zurückgegebene Ergebnis mit einer **WRITE**-Anweisung ausgeben.

Notizen