Übungen Ing.-Inf. – KE 6

Der C-Coding Styleguide ist einzuhalten!

Folgende Einstellungen sind für Debug und Release (All Configurations) vorzunehmen:

Einstellung	**Wert**
Solution Platform	x86
Properties->Conf. Properties->C/C++->General->Warning Level	Level4 (/W4)
Properties->Conf. Properties->C/C++->General->Treat Warnings As Errors	Yes (/WX)
Properties->Conf. Properties->C/C++->General->SDL checks	Yes (/sdl)
Properties->Conf. Properties->C/C++->Code Generation->Basic Runtime Checks	Default
Properties->Conf. Properties->C/C++->Code Generation->Security Checks	Enable Security Checks (/GS)

Solution muss in Debug und Release fehlerfrei kompilierbar sein.

Packen Sie das gesamte Verzeichnis der Solution in eine Zip-Datei und laden Sie diese in Moodle pünktlich hoch.

Aufgabe 1:
Implementieren Sie ein Programm, welches in main() drei lokale Variablen definiert (Datentyp sei int) und diese mit 73, 42 und 7 initialisiert.
1.1 Wo im Speicher (Adressen in hexadezimal) sind diese Variablen abgelegt? Tipp: Verwenden Sie printf(), den Formatbeschreiber %p und den Dereferenzierungsoperator &. Definieren Sie einen Zeiger piVal vom Datentyp int* und lassen Sie diesen auf die erste Variable zeigen.
1.2 Geben Sie mit printf die Adresse von piVal (Adresse in hexadezimal – Formatierer %p) auf dem Bildschirm aus.
1.3 Geben Sie mit printf den Wert von piVal (Hexcode – Formatierer %p) auf dem Bildschirm aus.
1.4 Geben Sie den Wert, auf den der Zeiger piVal zeigt (dezimal – Formatierer %d) auf dem Bildschirm aus.

Aufgabe 2:
Implementieren ein Programm, welches eine Struktur (struct Address, siehe KE06, Folie 11) deklariert. Definieren Sie in main eine Variable (sAddress) von diesem Typ unter Verwendung eines typedefs und initialisieren Sie die Variable bei der Definition. Definieren Sie anschließend einen Zeiger (psAddress) der auf sAddress zeigt. Verwenden Sie auch bei der Definition des Zeigers ein typedef.
3.1 Geben Sie die sAddress auf der Konsole aus.
3.2 Ändern Sie uiZipCode von sAddress (4711U) und geben Sie sAddress auf der Konsole aus.
3.3 Ändern Sie uiZipCode von sAddress (815U) mittels psAddress und geben Sie sAddress auf der Konsole aus.
Die Ausgabe von sAddress soll in einer Funktion erfolgen. Übergeben Sie sAddress mittels Call by Value.

Aufgabe 3:
Implementieren Sie ein Programm, welches den Unterschied zwischen "Call by Value" und "Call by Reference" für einfache und komplexe Datentypen aufzeigt. (siehe KE06, Folien 19-21). Sie können dazu den Programmcode von den Folien in Ihr Programm kopieren.

Übungen Ing.-Inf. – KE 6

Aufgabe 4:
Implementieren Sie ein Programm, welches ein Integer-Array int aiArray[3] = {0}; definiert und mit Null initialisiert. Arrays werden in C immer mit Call by Reference übergeben. Deklarieren und definieren Sie drei Funktionen, welche das Array übergeben bekommen. Jede Funktion nutzt dabei eine andere Möglichkeit um Arrays zu ergeben. Die erste Funktion soll eine 1 in das erste Element (Index 0) schreiben, die zweite Funktion soll eine 2 in das zweite Element (Index 1) schreiben und die dritte Funktion soll eine 3 in das dritte Element (Index 2) schreiben. Rufen Sie nun die drei Funktionen nacheinander auf und geben Sie anschließend das Array in main aus.
Rufen Sie anschließend eine der drei Funktionen erneut auf und übergeben Sie statt aiArray jetzt NULL. Was passiert? Schreiben Sie die Antwort als Kommentar oberhalb von main.
Implementieren Sie Sanity Checks in den drei Funktionen. Ist damit das obige Problem gelöst?