# **CPSC 240 Lecture on Arrays**

This work is licensed under the Creative Commons Attribution-ShareAlike 4.0 International License. To view a copy of this icense, visit http://creativecommons.org/licenses/by-sa/4.0/ or send a letter to Creative Commons, PO Box 1866, Mountain View, CA 94042, USA.

Non-legal summary in common language: You may copy and distribute this document. You may modify this document, and distribute the modified document. You must maintain the attribution to all authors of the original document and modifications thereof. Modified documents must be distributed under the original license. This is a 'free' document.

## Declarations of initialized arrays in segment data

Source code statements are appear in 10-point boldface font.

//Stores ascii value of each character of the string enclosed in quotes in the array.

welcome db "Welcome to fun programming.", 10, 0

//Stores each number in exactly one byte; the low order bits are stored; the high order bits are discarded. A warning will appear stating that the number greater than 255 will not fit in one byte, but the compilation will continue anyway.

myintegers1 db 10, -2, 7, -3, 15, 255, 256, 257, 258, 400

//Stores each number in exactly one quadword. There is no loss of bits if the stored number is in the range  $-2^{63}$  through  $+2^{63}-1$ .

myintegers8 dq 10, -2, 7, -3, 1023, 1024, 1025

//Stores each floating point number into one 8-bytes slot of the array

mydoubles8 dq 1.0, 2.0, 3.0, -2.5, -3.2, 6.0

//Stores each hex number into one 8-byte slot of the array

myhex8 dq 0x400400000000000, 0x40088000000000, 0x4008A0000000000

## Declarations of uninitialized arrays in segment bss

//The following four declarations create the exact same result, namely: an array of 80 bytes in size. The programmer can pick the one he or she prefers.

array1 resb 80 array1 resw 40 array1 resd 20 array1 resq 10

### Strings referenced in the text that follows

output\_float db "The float value from the array in base 10 is %1.18lf and in base 16 it is 0x%016lx.",10,0 output\_char db "The character from the array is %1c.", 10, 0 output\_integer db "The integer from the array is %014ld.",10,0

### X86 assembly statements that access arrays

;Copy the integer 3 into slot #2

mov qword [array1+2\*8], 3 ;qword is required

;Copy the integer 13 into slot #4

mov qword [array1+4\*8], 0X00000000000000D

;Attempt to copy the float number 3.5 into slot #5 Error: A floating point immediate value in base 10 is not allowed.

mov qword [array1+5\*8], 3.5

;Attempt to copy the float number 3.5 into any general register. Error: A floating point immediate value in base 10 is not allowed.

mov gword r13, 3.5 ;Error

;Attempt to push the float number 3.5 onto the system stack. Error: A floating point immediate value in base 10 is not allowed.

push 3.5

;Error

;Copy the immediate value 3.5 in hex format into slot #5 of the array. This is a syntax error.

mov qword [array1+5\*8], 0x400480000000000

:Error

;Copy the immediate value for 3.5 in hex first to a general register and then copy it from there to slot number 5 of the array. This is successful

mov qword r15, 0x4004800000000000

;Success

mov qword [array1+5\*8], r15

;Success

;Copy the ascii value of 'A' into byte number 2 of slot #6

mov byte [array1+6\*8+2], "A"

;Requires double quotes

;Copy the float number 0x400444000000000 into slot #1.

mov qword [array1+1\*8], 0x4004440000000000

;Copy the float number 0x4004440000000000 into slot #1: assembles without error, but does not make the copy correctly.

mov qword [array1+1\*8], 0x4004440000000000

;Copy the immediate value for 3.5 in hex first to a general register and then copy it from there to slot number 5 of the array. This is successful.

Output the float number in slot #5 of the array.

movsd xmm0, [array1+5\*8] mov rax, 1 mov rdi, output\_float mov qword rsi, [array1+5\*8] call printf

Output the single character value in slot #6.

mov qword rax, 0 mov rdi, output\_char mov qword rsi, 0 mov byte rsi, [array1+6\*8+2] mov rdx, rsi call printf

;Output the integer value in slot #2

mov qword rax, 0 mov rdi, output\_integer mov qword rsi, [array1+2\*8] call printf

This document intends to clarify the CPSC 240 lecture of February 15, 2018. There is never enough time to explain it all in single hour.

Date of last update February 17, 2018