Array of Struct

60 mins: 88

Time to get 100: 75mins

; ****Your name goes here*******

;Andrew Kirk

; This is Exam2_ArrayofStruct

; EE319K Fall 2014 exam2, November 6, 2014

; You edit this file only

```
    AREA   Data, ALIGN=4



    AREA    |.text|, CODE, READONLY, ALIGN=2
    THUMB
```

;****************** Linear*****************************

; Calculate the result of a linear equation y = 16*x-50

; Input parameter: x is unsigned 8 bits

; Output parameter: y is unsigned 8 bits

; Error conditions: implement ceiling on overflow

; implement floor on underflow

; Test Cases as (Input, Output) pairs:

; (0,0),(3,0),(4,14),(5,30),(11,126),

; (15,190),(19,254),(20,255),(100,255),(255,255)

; C prototype uint8_t Linear(uint8_t x){

```
    EXPORT Linear
Linear
  ; put your answer here
```

```
      MOV R2, #4
            CMP R0, R2
            BCC under
            LSL R0, #4 ;multiply by 16
            MOV R1, #50
            SUB R0, R0, R1
            MOV R2, #255
            CMP R0, R2
            BCS upper
            B done
under  MOV R0, #0
            B done
upper  MOV R0, #255
done   ; replace this line with your solution
      BX  LR

;******************Swap**************************
; You are given an 11-element 16-bit array.
; Your function should swap the order of the data in the array
; Input: pointer to array
; Output: none
; Error conditions: none
; Test Cases:
; 1. buf before: -5, 4, 7, 0,-1, 3, 4,-8, 2, 9, 9
;    buf after:  9, 9, 2,-8, 4, 3,-1, 0, 7, 4,-5
; 2. buf before: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10,11
;    buf after:  11,10, 9, 8, 7, 6, 5, 4, 3, 2, 1
; 3. buf before: 1000,2,3,4,5,-1000,7,10000,9,10,0
```

```
;    buf after:  0,10,9,10000,7,-1000,5,4,3,2,1000
; C prototype   void Swap(int16_t buf[11]){
      EXPORT Swap
Swap
; put your answer here
      ADD R1, R0, #20;
                              LDRH R2, [R0]
      LDRH R3, [R1]
            STRH R3, [R0]
            STRH R2, [R1]
            ADD R0, #2
            SUB R1, #2
                  LDRH R2, [R0]
      LDRH R3, [R1]
            STRH R3, [R0]
            STRH R2, [R1]
            ADD R0, #2
            SUB R1, #2
                  LDRH R2, [R0]
      LDRH R3, [R1]
            STRH R3, [R0]
            STRH R2, [R1]
            ADD R0, #2
            SUB R1, #2
                              LDRH R2, [R0]
      LDRH R3, [R1]
            STRH R3, [R0]
            STRH R2, [R1]
```

```
        ADD R0, #2
        SUB R1, #2
                LDRH R2, [R0]
    LDRH R3, [R1]
        STRH R3, [R0]
        STRH R2, [R1]
        ADD R0, #2
        SUB R1, #2



    BX  LR


;struct LabGrades{
;  int32_t size;
;  int32_t score[10];
;};
;typedef struct LabGrades LabGrades_t;
; ********************Average*******************
; You will write this function.
; The function should take a pointer to a lab grade structure and return the average.
; Average can be calculated only if the size is 1 to 10.
; Input:  Pointer to a lab grade structure
; Output: Average of the lab grades
; Error conditions: If the size is outside the range of 1 to 10, return 0.
; test data
; size  score                       Average
```

```
; 10  |  90  90  90  90  90  90  90  90  90  90 | 90
; 5   |  90  91  92  93  94                    | 92
; 1   | 100                                    |100
; 5   |  85  90  100 70 -25                    | 64
; 8   |  -4  -5  -6  -7 -10   1   2   5        | -3
; 0   |                                        |  0
; 255 |                                        |  0
; C prototype   int32_t Average(LabGrades_t *pt){ ; debug this code
      EXPORT Average
Average
  ; put your answer here
      PUSH {R4,R5}
            LDR R1, [R0] ;store the size of array
            CMP R1, #1
            BCC outrange
            CMP R1, #11
            BCS outrange

            ADD R2, R1, #0; establish a counter
            MOV R4, #0; sum register initialized to 0
nextval     ADD R0, #4
            LDR R3, [R0];get value from array
            ADD R4, R4, R3;sum
            SUBS R2, #1;decrement counter
            BNE nextval  ;if counter is not zero get the next value
      SDIV R0, R4, R1 ;get average
            B done1
outrange
```

```
            MOV R0, #0

done1   POP {R4,R5}             ; replace this line with your solution
      BX  LR

; **************ClassAverage****************
;   Find the average of all the lab grades in the class
;   Sum up all grades and divide by the number of grades
;   Do not sum up student averages and divide by the number of students
;   if size is 255, it means end of list
;   When dividing, do not round, simply divide sum/count
; Each Labgrade structure is 44 bytes (4 bytes for size and 40 bytes for 10 grades)
; Input:  array of Grades_t data
; Output: the average lab grade
; Error conditions: if there are no students or no grades, return 0
;-----------------------------------------------------------------
;Case 1: six students in the class
;{{5,{84,90,88,70,-25}},
; {1,{70}},
; {9,{90,90,90,90,-90,70,10,10,10}},
; {0,{}},
; {10,{80,80,80,80,80,80,80,80,80,99}},
; {2,{80,82}},
; {255,{}}
;}
;Class Average = 64; (see handout for explanation)
;-----------------------------------------------------------------
;Case 2: three students in the class
```

```
;{{2,{100,100,}},
; {1,{95}},
; {2,{90,90}},
; {255,{0}}
;}
;Class Average = (100+100+95+90+90)/5 = 475/5 = 95
;----------------------------------------------------------------
;Case 3: one student in the class
;{{4    ,{-1,-1,-1,-1}},
; {255,{0}}
;}
;Class Average = -1;
;
;Case 4:    no students at all
;{{255,{0}}
;}
;Class Average = 0;
;----------------------------------------------------------------
; C prototype   int32_t ClassAverage(LabGrades_t ee319k[]){
      EXPORT ClassAverage
ClassAverage
  ; put your answer here
      PUSH {R4-R7}
            LDR R1, [R0]
            CMP R1, #255
            BEQ zero
            MOV R4, #0; initialize sum
            MOV R7, #0; initialize grade counter
```

```
nextstud  LDR R5, [R0]; get number of grades
      ADD R2, R5, #0; make grade counter
                CMP R2, #0
                BEQ nextstudd
      CMP R5, #255
                BEQ avg

nextvalue       ADD R0, #4
          LDR R3, [R0];get value from array
          ADD R4, R4, R3;sum

          SUBS R2, #1;decrement counter
          BNE nextvalue  ;if counter is not zero get the next value
nextstudd     MOV R6, #11
          SUB R6, R6, R5; how far you have to go to get next number of grades
          LSL R6, #2; correct number of spaces to move pointer
          ADD R0, R0, R6; move pointer
          ADD R7, R5, R7; keeps track of number of grades
          B nextstud
avg   SDIV R0, R4, R7; divide sum of grades by number of grades
     B done2

zero
     MOV R0,#0 ; replace this line with your solution
done2  POP {R4-R7}
     BX  LR
```

END

Database

60 mins: 70

Time to get 100: 85mins

```
/*
;*****Your name goes here*******
; -5 points if you do not add your name

;This is Exam2_DataBase
;EE319K Practice exam
;You edit this file only
*/
#include <stdint.h>
/*
****************** Size*****************************
 Determines the length of a null-terminated string.
 Input parameter: pt points to variable-length string
 sentinel is 0
 Output parameter: Return the length of string
 Error conditions: if string is empty, return 0
Test cases
"Jony"                size = 4
"Ramesh Yerraballi"    size = 17
{4,1,2,3,0}            size = 4,
"Jonathan Valvano"     size = 16
```

```
""                  size = 0
*/
uint32_t Size(const char *string){
// put your code here
        int32_t length=0;
                while( *string != 0){
                        length ++;
                        string ++;
                }

  return(length);  // change this line
        }
/****************** Average ****************************
Find the average of the data in a variable length buffer
The data in the buffer range from -127 to +127
Input: size is the number of elements in the buffer
      pt is a pointer to the buffer
Output: average of the data in the buffer
Error conditions: Return 0 if the buffer is empty.
Test cases
 size value buffer
  5,   {1,4,3,3,4}                //Average=3
 10,   {-3,-10,0,0,0,0,0,1,2,0,0}    //Average=-1
  5,   {-1,2,3,-3,4}              //Average=1
  7,   {-6,-9,-7,-8,-8,-9,-9}      //Average=-8
  0,   {}                        //Average=0
*/
int8_t Average(uint32_t size, const int8_t *pt){
```

```
// put your code here
int32_t sum=0, average=0, i;
        if (size!=0){
                        for (i=0; i<size; i++){
                                sum= sum + *pt;
                                pt++;
                        }
                        average=sum/(int32_t)size;
                }
        else{
                average=0;
        }
  return(average);  // change this line
}
struct student{
  char name[20];
  int8_t data[8]; // skip if data=-128
};
typedef struct student student_t;

/* ****************** Grade ****************************
Finds the average student grade
Skip grades that are -128
Input: one student record
Error conditions: Return -128 if there are no grades in the list.
Test cases
   "Jony",   { 100, 100, 100, 100,-128,-128,-128,-128}  //Grade=100
   "Ramesh", {  90,  90,  90,  90,-128,  90,-128,  90}  //Grade=90
```

```
  "Mattan", {  95, 100,  95,  90,-128,  80,-128, 110}  //Grade=95
  "Woosek", { 110, 100, 100, 100, 100,  98, 100, 100}  //Grade=101
  "",       {-128,-128,-128,-128,-128,-128,-128,-128}  //Grade=-128
*/
int8_t Grade(student_t s){
// put your code here
int8_t *gradept=0, length1=0;
int32_t sum=0, average=0, i;
        gradept=s.data;

        for( i=0; i<8;i++){
                if(*gradept!=-128){
                  length1 ++;
                        gradept ++;
                }
                else{
                        gradept++;
                }
        }
        gradept=s.data;

        if (length1!=0){
                        for (i=0; i<8; i++){

                                if (*gradept!= -128){
                                sum= sum + *gradept;
                                gradept++;}
```

```
                                else{
                                        gradept++;}
                        }
                        average=sum/(int32_t)length1;
                }
        else{
                average=-128;
        }
 return(average);  // change this line
}
```

/* ****************** TopStudent ****************************

Find the student with the highest average

Input: Array of 10 students

Output: index value of the student with highest average

Error conditions: none

Test case

```
  "Jony",     { 100, 100, 100, 100,-128,-128,-128,-128}  //Grade=100
  "Ramesh",   {  90,  90,  90,  90,-128,  90,-128,  90}  //Grade=90
  "Mattan",   {  95, 100,  95,  90,-128,  80,-128, 110}  //Grade=95
  "Ben",      { 110,-128,-128,-128,-128,   0,-128,-128}  //Grade=55
  "Sourabh",  { 100,-128,-128,-128,-128,  90,-128,-128}  //Grade=95
  "Dylan",    {  80,-128,-128,-128,-128,  80,-128,-128}  //Grade=80
  "Kei",      {  90,-128,-128,-128,-128,  90,-128,-128}  //Grade=90
  "Sean",     {  99,  99,  99,  99,-128,  99,-128,-128}  //Grade=99
  "Nagaraja", {  98,-128,-128,-128,-128,  98,-128,-128}  //Grade=98
  "Woosek",   { 110, 100, 100, 100, 100,  98, 100, 100}  //Grade=101 <-return 9
```

```
*/
uint32_t TopStudent(const student_t EE319K[10]){
// put your code here
        int i=0, max=Grade(EE319K[0]), maxindex=0;
  for(i=0; i<10; i++){
                student_t element= EE319K[i];
        int average=Grade(element);
                if( max<average){
                        max=average;
                        maxindex=i;
                }
        }



        return(maxindex);  // change this line
}
```