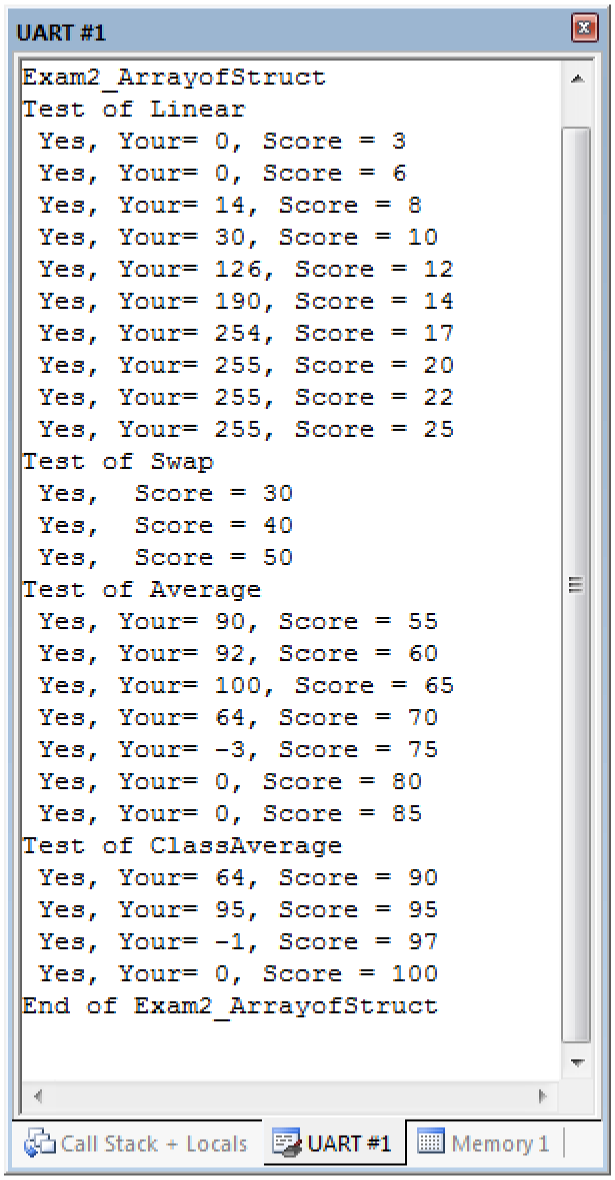
EID:____________________ First:___________________ Last:____________________________

Scoring The correct output values are shown in the figure. Your grade will be based both on the numerical results returned by your program and on your programming style. In particular, write code that is easy to understand, easy to debug, easy to change. Please employ good labels, pretty structure, and good comments.

Performance Score= Run by TA		TA:



```
Exam2_ArrayofStruct
Test of Linear
 Yes, Your= 0, Score = 3
 Yes, Your= 0, Score = 6
 Yes, Your= 14, Score = 8
 Yes, Your= 30, Score = 10
 Yes, Your= 126, Score = 12
 Yes, Your= 190, Score = 14
 Yes, Your= 254, Score = 17
 Yes, Your= 255, Score = 20
 Yes, Your= 255, Score = 22
 Yes, Your= 255, Score = 25
Test of Swap
 Yes,  Score = 30
 Yes,  Score = 40
 Yes,  Score = 50
Test of Average
 Yes, Your= 90, Score = 55
 Yes, Your= 92, Score = 60
 Yes, Your= 100, Score = 65
 Yes, Your= 64, Score = 70
 Yes, Your= -3, Score = 75
 Yes, Your= 0, Score = 80
 Yes, Your= 0, Score = 85
Test of ClassAverage
 Yes, Your= 64, Score = 90
 Yes, Your= 95, Score = 95
 Yes, Your= -1, Score = 97
 Yes, Your= 0, Score = 100
End of Exam2_ArrayofStruct
```

I promise to follow these rules

This is a closed book exam. You must develop the software solution using the **Keil uVision** simulator. You have 70 minutes, so allocate your time accordingly. You must bring a laptop and are allowed to bring only some pens and pencils (no books, cell phones, hats, disks, CDs, or notes). You will have to leave other materials up front. Each person works alone (no groups). You have full access to **Keil uVision**, with the **Keil uVision** help. You may use the Window's calculator. You sit in front of a computer. You edit, assemble, run, and debug the assignment. You do NOT have access the book, internet or manuals. You may not run out of DropBox, GoogleDrive, Box or similar backup service during the exam. You may not take this paper, scratch paper, or rough drafts out of the room. You may not access your network drive or the internet. You are not allowed to discuss this exam with other EE319K students until Friday afternoon.

The following activities occurring during the exam will be considered scholastic dishonesty:

1) running any program from the PC other than **Keil uVision**, or a calculator,
2) communicating with **anyone else** except for the instructors **by any means** about this exam until Friday
3) using material/equipment other than a pen/pencil,
4) hard-coding so it outputs answers that give points without actually solving the problem,
5) modifying anything other than **Exam2.s**

Students caught cheating will be turned to the Dean of Students. Your signature is your promise that you have not cheated and will not cheat on this exam, nor will you help others to cheat on this exam:

Signed: ____ November 4, 2014

Procedure

First, you will log onto the computer and download files from the web as instructed by the TAs.

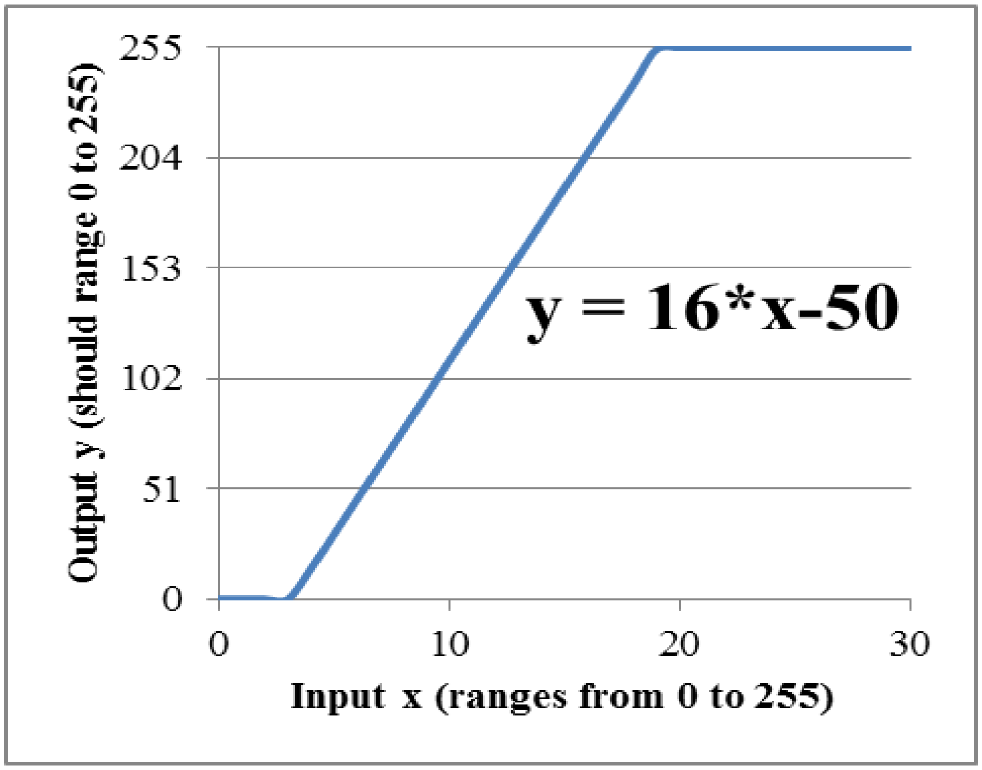
Web site	xxxxxxxxx
User:	yyyyyyyyy
Password:	zzzzzzzzzz

UNZIP the folder placing it **ON THE DESKTOP**. You are not allowed to archive this exam during the exam time. Within **Keil uVision** open the project, put your name on the first comment line of the file **Exam2.s**. Before writing any code, please build and run the system. You should get output like the figure above (but a much lower score). You are allowed to create backup versions of your program. If you wish to roll back to a previous version, simply open one of the backup versions.

My main program will call your subroutines multiple times, and will give your solution a performance score of 0 to 100. *You should not modify my main program or my example data.* Each time you add a block of code, you should run my main program, which will output the results to the **`UART#1`** window. After you are finished, raise your hand and wait for a TA. The TA will direct you on how to complete the submission formalities. The TA will run your program in front of you and record your performance score on your exam cover sheet. The scoring page will not be returned to you.

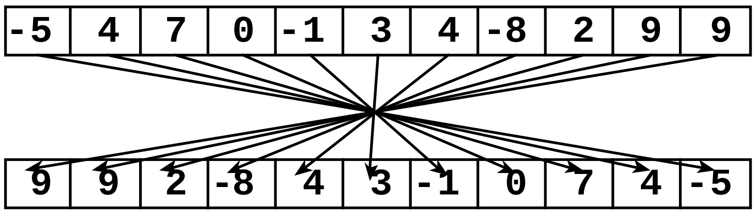
The exam has four parts a) through d), details of which are given in the starter code (Exam2.s):

Part a) Write an assembly function, called **`Linear`**, with input x, and output y. The function will calculate the output of a linear function, y = 16*x-50. You can assume the input will be an 8-bit unsigned integer (0 to 255). However, you must generate an 8-bit output also in the range of 0 to 255. Set the output to the ceiling (255), if the calculation yields an overflow in 8-bit math. Set the output to the floor (0), if the calculation yields an underflow in 8-bit math.



x	y
0	0
1	0
2	0
3	0
4	14
5	30
6	46
7	62
8	78
9	94
10	110
11	126
12	142
13	158
14	174
15	190
16	206
17	222
18	238
19	254
20	255
21	255
22	255
23	255

Part b) In this question you will write an assembly function, called **`Swap`**, that swaps the elements of a fixed-length array in place. You are passed a pointer to an array of signed 16-bit numbers. There are exactly 11 elements. Your function should reverse the data in place. I.e., read and write into the same array.

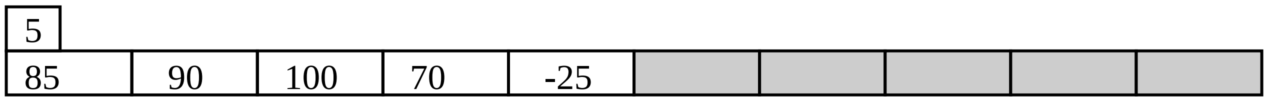


Part c) You will write the function **`Average`**. In parts c) and d), we use a structure to define a variable-length array. One element will be the size and the second element will be the array data. There will be a variable number of elements stored in the array, but 10 elements are always allocated. If the size is outside the range of 1 to 10, the function **`Average`** should return 0. The data are signed 32-bit integers. Each structure has exactly 44 bytes.

```
struct LabGrades{
  int32_t  size;
  int32_t score[10];
}
typedef struct LabGrades LabGrades_t
```

For example, assume Jony has completed 5 labs; we can define a variable-length array to store his grades this way:

```
LabGrades_t Jony = { 5, {85,90,100,70,-25}};
```



The function **`Average`** could be called this way

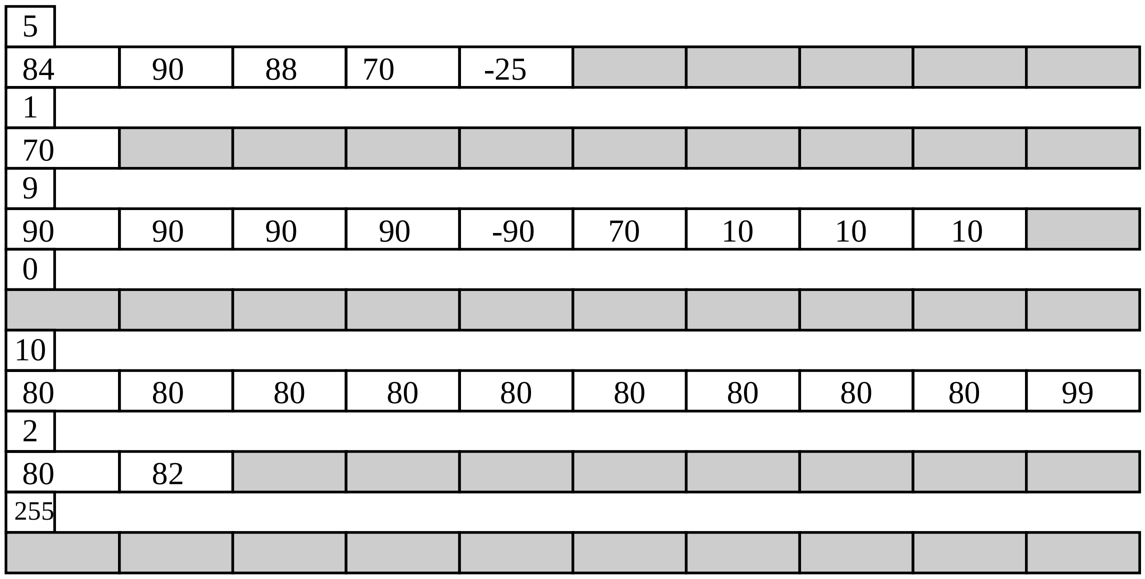
```
 int32_t JonyAve;
 JonyAve = Average(&Jony);
```

This should set **`JonyAve`** = (85+90+100+70 + -25)/5 = 320/5 = 64

Write and debug this function that takes a pointer to a lab grade structure and is supposed to return the average.

Part d) We now extend this problem to define the entire EE319K class by making an array of structures. You will write an assembly function, called `ClassAverage`, which returns the average of all grades in the class. To handle the variable number of students, we will place a sentinel entry with a size of 255 at the end of the list. One possibility with 6 students is shown below. Each student could have any number of grades defined from 0 to 10. If the size is -1, then it means end of data. The entry at size equals -1 is not considered data.

```
LabGrades_t EE319K[7] = {
  { 5,  {84,90,88,70,-25}},
  { 1,  {70}},
  { 9,  {90,90,90,90,-90,70,10,10,10}},
  { 0,  {0}},
  { 10, {80,80,80,80,80,80,80,80,80,99}},
  { 2,  {80,82}},
  {255, {0}}                             // end of list
};
```



In this case there are 27 grades and the function `ClassAverage` should return 64 (do not round).
(84 + 90 + 88 + 70 + -25 + 70 + 90 + 90 + 90 + 90 + -90 + 70 + 10 + 10 + 10 + 80 + 80 + 80 + 80 + 80 + 80 + 80 + 80 + 80 + 99 + 80 + 82)/27 = 1728/27 = 64

Important Notes:

- Your functions should work for all cases given to it by the grader.
- Handle the simple cases first and the special cases last.
- ***This exam is different from other exams. You should NOT call your function in part c) from within your function in part d).***

Submission Guidelines:

- Log onto Canvas and submit your **Exam2.s** source file into the Exam2 submission link. Be careful because only one submission will be allowed.