Computer Architectures **02LSEOV 02LSEOQ [M-Z]**	Delivery date: Thursday 6/12
Laboratory **7**	Expected delivery of lab_07.zip must include: • zipped project folder of the exercises 1 and 2 • this document compiled possibly in pdf format.

Solve the following 2 problems by starting from the *template.s* file.

Exercise 1) The owner of a shop is restocking shelves. He made a list with items to buy; the items are identified by means of a code. He wants to know the total amount of the expense according to the price list of his wholesaler. Write a program in ARM assembly language to compute the amount of the expense according to the following details.

The price list of the wholesaler is a table where each entry consists of two integer values: the identification code of the product (4 bytes) and the price (4 bytes). The table is ordered according to the identification code. For example:

```
Price_list DCD 0x004, 120, 0x006, 315, 0x007, 1210, 0x00A, 245
           DCD 0x010, 228, 0x012, 7, 0x016, 722, 0x017, 1217
           DCD 0x018, 138, 0x01A, 2222, 0x01B, 34, 0x01E, 11
           DCD 0x022, 223, 0x023, 1249, 0x025, 240, 0x027, 112
           DCD 0x02C, 2245, 0x02D, 410, 0x031, 840, 0x033, 945
           DCD 0x036, 3211, 0x039, 112, 0x03C, 719, 0x03E, 661
           DCD 0x042, 230, 0x045, 1112, 0x047, 2627 , 0x04A,
           265
```

The list of items to buy is stored in another pool with two integer values for each entry: the identification of the product (4 bytes) and the desired quantity (4 bytes). The table is not ordered. For example:

```
Item_list DCD 0x022, 14, 0x006, 431, 0x03E, 1210, 0x017, 56342
```

A binary search should be implemented for searching the price of the items in the `Price_list` table. The C code of the binary search is:

```
first = 0
last = num_entries - 1;
index = 0;
while (first <= last)
      {
      middle = (first + last) / 2;
      if (key == table[middle])
      {
            /* element found */
            index = middle;
            break;
      }
      else
```

```
            if (key < table[middle])
                 last = middle - 1;
            else
                 first = middle + 1;
}
```

At the end of the program, register r10 should contain the amount of the expense if all products are present in the price list, or 0 if at least one product is missing in the price list.

Exercise 2) Create a new project by starting from the previous exercise and suppose that the entries in the price list are not sorted (unsort on purpose the list of values you have used in exercise 1); you have to implement first a sorting algorithm and then to perform the same calculation function of exercise 1).

Use the sorting algorithm you prefer to reorder the list of values (i.e., use algorithms you studied in previous courses or search on the web).

Remember that:

- the sort routine must sort both the key and the data associated with each entry
- it is not possible to reorder the content of a portion of memory in a read-only section, but a proper area needs to be allocate in a read-write section and used to store/order the list.

Report the selected values in the table below.

	Execution time	Code size	Data size
Exercise 1)	123,45 us	554	204
Exercise 2)	2,63225 ms	568	764

Respond to the following open questions.

Q1: what section have you used to store the ordered sequence?

I've used the Heap

Q2: Which address range is assigned to this section and which memory of the system is used?

The address range is from 0x10000000 to 0x10000200
The memory used is the Code memory that goes from 0x00000000 to

0x1FFFFFFF

Q3: describe what is the worst case of initial order (the one who takes the most to sort) and motivate.

The worst case for my algorithm (bubble sort) is the one where the sequence of products is ordered in descending order because this involves the execution of ~n^2 swaps.