Eurovision 2022 in Turin!

Exercise 1)

The **Eurovision Song Contest** 2022 will be held at the PalaOlimpico in **Turin**, Italy, following the country's victory at the 2021 contest with the song "Zitti e buoni" by *Måneskin*.

Write a program in **ARM assembly** language to <u>manage the sale of contest tickets</u>. PalaOlimpico is organized in different sectors with different prices. For example, you have the following lists <u>to be initialized in a pool</u>:

```
Sector_prices DCD 0x01, 25, 0x02, 40, 0x03, 55, 0x04, 65, 0x05, 80
              DCD 0x06, 110

Sector_quantity DCD 0x02, 250, 0x05, 250, 0x03, 550, 0x01, 150, 0x04,
                DCD 100, 0x06, 200

Num_sectors DCB 6
```

Sector_prices is a table where each entry consists of two integer values: the ID of the sector (4 bytes) and the price of each ticket in that sector (4 bytes).

Sector_quantity is a table where each entry consists of two integer values: the ID of the sector (4 bytes) and the number of places available in that sector (4 bytes).

Num_sectors is a 1 byte constant and indicates the number of sectors available in PalaOlimpico.

Write a program to respond to a purchase request.

The <u>request</u> is stored in the following pool, where you have a set of 2 items: the sector ID (hexadecimal) and the whished quantity. The variable `Tickets_requests` stores the amount of ticket requests.

For instance, in the following example the user wishes to buy tickets from three different sectors:

```
Tickets DCD 0x05, 2, 0x03, 10, 0x01, 120
Ticket_requests DCD 3
```

If the tickets are available, update the `Sector_quantity` and store the total cost of the purchase in a 4-byte variable stored in RAM, named *total_tickets*; otherwise, if the sector is sold out (or the desired quantity is not available), store zero in the same variable, and 0x01 in R11 to underline that the procedure has not been completed.

Moreover, if the desired number of tickets is greater than 10, apply a 50% group discount for **Black Friday.**

Exercise 2) Experiment the SVC instruction.

Write an ARM assembly program that invokes an `SVC` instruction when running a <u>user routine</u> with <u>unprivileged access level</u>.

By means of invoking a SuperVisor Call, we want to implement the squared power (x^2) or the **integer approximation of the square root** of a number ($\lfloor\sqrt{x}\rfloor$). The approximation of the square root is calculated by iterating on all natural numbers n in ascending order until the following condition is satisfied: $n^2 \leq x$. The value of **x** is stored in R0.

The SVC instruction is encoded as follows:
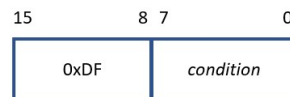
- Bits [15:8]: Opcode of the SVC Assembly ARM Instruction
- Bits [7:0]: This field indicates the operation that must be performed, according to its content.
  - If the content is equal to 0, the squared power must be done,
  - Else if the content is equal to 1, the integer approximation of the square root must be done.
  - Else, NOP operations.

> **Example:** SVC 1 and R0=0x11
> Your algorithm must return 4.

The result of your code must be saved in the PSP and returned as specified in the figure below. Then, outside the *SVC_handler*, save the result in a 4-byte variable *SQResult*.

> Q1: Describe how the stack structure is used by your project and which stack you are using and why.
>
> Nel progetto uso uno stack full descending in linea con lo standard ARM. Prima di eseguire la chiamata SVC decremento SP di 8 byte in quanto lo stack è allineato su quella dimensione. In questo modo andrò a riservare uno spazio di memoria in cui memorizzare il risultato dell'operazione. Senza questo passaggio andrei a salvarlo all'interno di una sezione di memoria non dedicata allo stack unprivileged. All'interno della funzione andrò ad utilizzare solamente 4 degli 8 byte riservati per il salvataggio del risultato. All'uscita dell funzione vado a reincrementare SP di 8 byte per riportarlo al valore iniziale.

Q2: What needs to be changed in the `SVC` handler if the access level of the caller is privileged? Please report the code chunk that satisfies this request.

Se SVC viene chiamato dalla modalità privileged andrò a sostituire la funzione MSR Rn, PSP con MOV Rn, SP. Questo perchè si userà lo stesso stack pointer all'interno dell'handler, per cui non ho bisogno di andare a recuperare lo stack frame dallo stack dedicato alla modalità unprivileged.