Objectives:
1. Practice with index addressing mode, branch instructions and the usage of subroutine.
2. Learn how to implement reentrant functions in assembly language through the use of factorials.

Submissions:
Follow the same previous guidelines. That is, including: introduction, implementation, description, flowchart, pseudo-code, results, conclusion, .LST file, snapshots, etc.

Lab Description:
Given an array of unsigned numbers, you need to write and test a program to calculate the factorial of each number using subroutine. Note that the output must always fit in one byte. The end of the array is designated with the number $FF, so your program should stop executing and exit the loop once it encounters the number $FF. In this lab, you need to use a nested loop structure to operate each number in the array. IMPORTANT! You are NOT allowed to use the “count” method for this nested loop; otherwise, marks will be deducted.

Your main program should call the subroutine to calculate factorial for each element in the input array. The subroutine will then calculate the factorial of the inputed number in a RECURSIVE way and return the calculated factorial (or return error in the form of a carry flag set).

Requirements:
Please follow the requirements below to avoid any deduction of your lab score:
- The program must begin at $C000.
- The stack pointer must be initialized to $DFFF.
- The data segment must begin at $D000.
- Run your program for the following two cases:
  - DATA FCB 0,5,2,1,4,3,$FF
  - DATA FCB 0,5,2,6,3,$FF
- The result must be stored in RESULT.
- End your program with these statements:
  - DONE BRA DONE Infinite loop
  - END

Useful Information:
Remember that if F(n) is the Factorial of n, then Factorial can be expressed recursively as:

\[ F(1) = 1 \]
\[ F(n) = F(n-1) \times n \]


```plaintext
Sub Routine: fact(n)

Flow Chart:

1. Return
2. Return Error
3. n = n * tmp
4. Call fact(n-1)
5. Return

Table:

<table>
<thead>
<tr>
<th>n</th>
<th>fact(n)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>24</td>
</tr>
<tr>
<td>5</td>
<td>120</td>
</tr>
<tr>
<td>6</td>
<td>720</td>
</tr>
</tbody>
</table>

Example: fact(5) = 120
```
`\text{Input: } A \rightarrow n`
MAIN

\[ R[i] = \text{fact}(D[i]) \]

\[ i = 1 + \]

\[ \text{END} \]

\[ C[i] = \text{flag} \]

\[ i = 0 \]

\[ \text{MAIN} \]
Note: fact(n)

Output: i = i - fact(n)
Input: A, n

Assembly: