Department of Mathematics **2025/2026**
Level: 1st year Mathematics

Final Exam: ASD2

Exercise 1 (5 pts) :

1. Define an enumeration named `GradeCategory` with the following values: `FAIL`, `PASS`, `GOOD`, and `EXCELLENT`.
2. Define a structure named `StudentModule` that contains:
   - `moduleName` (a character array).
   - `score` (a float).
3. Write a function `getCategory` that takes a `StudentModule` as an argument and returns the corresponding `GradeCategory` :
   - FAIL if score < 10.
   - PASS if 10 ≤ score < 12.
   - GOOD if 12 ≤ score < 15.
   - EXCELLENT if score ≥ 15.

Exercise 2 (5 pts) :

1. Write a program that reads integers from a text file named `numbers_list.txt`.
2. Count how many of these integers are even and how many are odd.
3. Write all the even numbers to a new text file named `even_numbers.txt` and all the odd numbers to `odd_numbers.txt`.
4. Finally, print the total count of even and odd numbers to the console.

Exercise 3 (5 pts) :

1. Write a function *`void incrementBy(int *value, int amount)`* that takes a pointer to an integer and an integer *`amount`*. The function should increment the value pointed to by *`value`* by *`amount`*.
2. In the main function:
   - Declare an integer variable *`counter`* and initialize it to 50 .
   - Declare an integer variable *`step`* and initialize it to 10 .
   - Call *`incrementBy`* to increase *`counter`* by *`step`* .
   - Print the value of *`counter`* before and after the function call to verify the change.

Exercise 4 (5 pts) :

Given the following structure for a singly linked list node:

```
typedef struct Node {
    int data;
    struct Node *next;
} Node;
```

1. Write a function `Node* insertAtEnd(Node* head, int newData)` that creates a new node with `newData` and inserts it at the end of the singly linked list. The function should return the head of the list.
2. In the main function:
   - Create an empty linked list.
   - Use `insertAtEnd` to add the values 10 , 20 , and 30 to the list.
   - Write a small loop to print all elements of the list to verify the insertion.
   - Free the allocated memory.

GOOD LUCK!