TU856/1 & TU858/1 Programming Assignment #2

Due Date: Sunday, March 7th, 2021 (23.59 GMT)

You are required to develop a security authentication program based on an access code. The access code is 4 single-digit integer numbers between 0-9. The program should allow a user to enter a code, encrypt the number and compare it to an authorised access code. The program should also allow the user to decrypt an already encrypted code.

When your program begins executing, the default authorised access code is **4523** (encrypted form of 1234 – see encryption algorithm below). This code must be stored in a 1-D array called *access_code* and should not be changed. You should use a different 1-D array to store the code entered by the user.

Your program should be menu-driven and must display the following simple menu when run:

1. Enter any code
2. Encrypt code entered and verify if correct (i.e., matches the authorised access code, i.e., 4523)
3. Decrypt code
4. Display the number of times the encrypted code entered matches the authorised code (i) Successfully (ii) Incorrectly
5. Exit Program

Note:

- Each menu option must be implemented in a separate function.
- All functions must pass parameters using **Pass by Reference**. Do NOT pass parameters using Pass by Value.
- All reading and writing to/from arrays must use pointer notation – do not use subscript notation (i.e., using [ ])

Requirements:

(Each implemented in a separate function):

1. Enter 4 single-digit integers. Perform any necessary validation (error-checking).

2. Encrypt the code entered. You should use the following algorithm to encrypt the 4 single-digit integers:

   Encryption Algorithm:
   - Swap the 1st number with the 3rd number.
   - Swap the 2nd number with the 4th number.
   - Add 1 to each number.
   - If any number has a value equal to 10, change this value to 0.

Compare the encrypted code with the access code (4523) stored in the 1-D array called *access_code*. If the two codes match, display "Correct Code entered". If the two codes do not match, display "Wrong Code entered" .

3. Decrypt an encrypted code. You should use the following algorithm to decrypt the encrypted code only:

   Decryption Algorithm:
   - Subtract 1 from each number.
   - If any number has a value equal to -1, change this value to 9.
   - Swap the 1st number with the 3rd number.
   - Swap the 2nd number with the 4th number.

4. Count and display the number of times the encrypted code entered matches the authorised code successfully & incorrectly during the current running program. If the program ends, reset all.

5. The program should terminate, i.e., end, gracefully.

Features to include:

- After each option has finished, your program should return to the main menu and allow the user to select another option.
- The user should only be allowed to encrypt their code (i.e., select option 2) if the code is NOT already encrypted.
- The user should only be allowed to decrypt their code (i.e., select option 3) if the code IS already encrypted.
- Only encrypted codes should be compared with the access-code (option 2).
- Display appropriate error messages to handle any errors.

Submission details:

1. Submission file name: **assignment2.c**
2. Submit your assignment2.c file on Brightspace. This must be submitted on or before Sunday, March 7th, 2021 (23.59 GMT).

Late submissions: Late submissions will incur a progressive penalty of 10% each day late, i.e., one day late loses 10%, two days late loses 20%, etc.,

No submissions accepted after 1 week and a zero-mark awarded.

NB - This is an individual assignment and **NOT** a group one. Do your own work and do not plagiarise your code. Anti-plagiarism software will be used to randomly check submissions. Any submitted code suspected of having been plagiarised will be brought to the attention of the module examiners for specialised checks under the TU Dublin general assessment regulations.

See marking scheme (rubric) below.

Marking scheme (Rubric):

Table 1 shows the marks allocated for this assignment.

Functionality (Requirements). Code should meet the highest professional standards	Menu option 1 (Enter code)	5%	
	Menu option 2 (Encrypt code and compare to valid code)	20%	
	Menu option 3 (Decrypt code)	15%	
	Menu option 4 (Count code entered (i) Successfully, (ii) Incorrectly)	20%	
	Menu option 5 (Program ends gracefully)	5%	
Error checking (Validation) Validate all program input, etc.,		15%	
		Sub Total:	**80%**
Commenting	Program Description, Author, Date	5%	
	Good comments throughout code body	10%	
Indentation	Correct and consistent indentation throughout code body	5%	
		Sub Total:	**20%**

Table 1: Marking scheme (Rubric)