Programme Code: DT211C, DT228, DT282
Module Code: CMPU1025
CRN: 22500, 22388, 26445

TECHNOLOGICAL UNIVERSITY DUBLIN

KEVIN STREET CAMPUS

BSc. (Honours) Degree in Computer Science (Infrastructure)

BSc. (Honours) Degree in Computer Science

BSc. (Honours) Degree in Computer Science (International)

Year 1

SEMESTER 2 EXAMINATIONS 2019/2020

Programming

Dr. Michael Collins
Dr. Deirdre Lillis

Follow all instructions precisely

All code to be submitted on Brightspace

Deadline Date: Friday, May 15th, 2020 (00.01am)

You are required to develop a program that will perform security authorisation based on numeric codes. The codes are four (4) single-digit integer numbers between 0-9. The program should allow a user to enter any four-digit code or generate a random code, encrypt the code and compare it to a fixed, 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 format of the number 1234 – see encryption algorithm below). This code must be stored in a 1-Dimensional array called *access_code* and should **not** be allowed to be changed.

Your program must be menu-driven and must display a simple menu when executed. The menu must include the following options:

1. Enter a code or generate a random code
2. Encrypt code
3. Check if the encrypted code matches the default authorised access code, i.e., 4523
4. Decrypt code
5. Display the number of times the encrypted code is:
   i. Correct
   ii. Wrong
6. 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** only – do not use subscript notation (i.e. using square braces) to access your array(s).

Requirements:

(Each following menu option must be implemented in a separate function):

1. The user must choose whether to enter any four (4) single-digit integers **or** generate a random code (0000 – 9999 inclusive). Perform any necessary validation (error-checking).
2. Encrypt the code entered. You should use the following algorithm to encrypt the 4 single-digit integer code:

   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.

3. Compare the encrypted code with the access code (4523) stored in the 1-Dimensional array called *access_code*. If the two codes match, display a message saying "Correct Code entered". If the two codes do not match, display a message saying "Wrong Code entered" .
4. Decrypt an encrypted code. You should use the following algorithm to decrypt an 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.

5. Using the following Structure template, count and display the number of times the encrypted code is correct and/or wrong in each run of the program:

```
struct code_counter {

    int correct_code;
    int wrong_code;
};
```

6. The program should terminate 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 input errors.

Submission details:

1. Submit your program using Brightspace. This must be submitted on or before Friday, May 15th, 2020 (00.01am).
2. Submission File name: **exam.c**
3. Submit Declaration of own work: **Declaration.docx**

Note: Anti-plagiarism software will be used to check every code file submission. Any code suspected of having been plagiarised will be brought to the attention of the module examiners for specialised checks. In cases where evidence of plagiarism is detected, the matter will be brought to the attention of TU Dublin management and the Examination Board.

Marking scheme (Rubric):

Table 1 shows the marks allocated for this assignment.

Functionality (Requirements). Includes use of functions, parameter passing – pass by reference, pointer-notation accessing arrays, using Structure, using switch statement where appropriate, use of symbolic names, etc.,	Option 1 (Code entry: manual or random generated)	15%	
	Option 2 (Encrypt code)	10%	
	Option 3 (Encrypted code comparison)	5%	
	Option 4 (Decrypt code)	10%	
	Option 5 (Counter for correct & wrong code entry)	20%	
	Option 6 (Graceful termination)	5%	
Error checking (Validation) See "features to include" above, validate all program input, etc.,		15%	
		Sub Total:	**80%**
Commenting	Program Description, Author, Date	5%	
	Comments throughout code body	10%	
Indentation	Correct and consistent indentation throughout code body	5%	
		Sub Total:	**20%**

Table 1: Marking scheme (Rubric)