Practical 3 (Week 4)

The learning outcomes of this practical include:

- Learn to use arrays to store data
- Understand and learn to apply linear search
- Understand how to pass an array to a function
- Prepare knowledge for assignment 1.

Task 3.1: Download the code "`task3._1.cpp`" from vUWS. The code is to search a target number in an array. However, the code can only display the index of its first occurrence in the array if it is found and ignore all the rest occurrences. Change the code so that it can found all the positions of the target number in the array. For instance, assume that the list of numbers is:

4, 2, 5, 8, 3, 2, 7, 6, -1

If the target is 2, the code should display 1 and 5.

Task 3.2: Download program "`task3_2.cpp`" from vUWS. This program creates a two-dimensional array, named `table`, fills the array with random numbers: `-1`, `0`, or `1`, and display the content of the array. Extend the program so that it can take an input of row and column numbers, display the value in that cell as well as all values next to the cell (up, down, left and right).

Hint:

1. *If the user's input is the actual row and column numbers, you need to convert it to the indices of the array to access the data.* For instance, if the content of the table is the following:

   ```
   1  0 -1  1
   0  1  0 -1
   0 -1  1  0
   ```

   and the user's input is (2, 4), the indices of the cell in the two-dimensional array are (1,3) and the value of that cell is then -1.

2. *There are up to four neighbours of each cell, depending on its position in the table: the corners, edges or inside.* For instance, based on the user's input (2, 4), the values in the neighbour cells are 1 (up), 0 (down), 0 (left). There is no number on the right.

   ```
   1  0 -1  1
   0  1  0 -1
   0 -1  1  0
   ```

Task 3.3: Download the code *TicTacToeFunctional.cpp* from vUWS. The program partly implemented the game of TicTacToe, which allows a user to input the coordinates of each move and display the move on the board. It is runnable but one of the functions, *isValidMove()*, has not been implemented thus the program can't check if an input is valid or not. Add your code to complete this function so that the program can check if an input of coordinates is in the right range and the corresponding cell is not occupied.

Task 3.4: Write a program for airplane seat reservation. Assume that there is a small airplane that has 32 seats in 8 rows. Each row has 4 seats.

```
  1 2 3 4
1 _ _ _ _
2 _ _ _ _
3 _ _ _ _
4 _ _ _ _
5 _ _ _ _
6 _ _ _ _
7 _ _ _ _
8 _ _ _ _
```

A customer can book a seat by providing the seat's row and column numbers. For instance, if a customer books the seat (2,4), then the seat in row 2 and column 4 is occupied:

```
  1 2 3 4
1 _ _ _ _
2 _ _ _ X
3 _ _ _ _
4 _ _ _ _
5 _ _ _ _
6 _ _ _ _
7 _ _ _ _
8 _ _ _ _
```

If another customer books the seat (7,3), then the seat availability becomes:

```
  1 2 3 4
1 _ _ _ _
2 _ _ _ X
3 _ _ _ _
4 _ _ _ _
5 _ _ _ _
6 _ _ _ _
7 _ _ X _
8 _ _ _ _
```

Write a program that can display availability of airplane seats (in the format shown above), take customer's booking of seats and update seat availability after taking customer's booking. If a customer requests in a seat that is already reserved, your program should say that that seat is occupied and ask for another choice. Your program should keep running until no further booking or all seats are reserved.