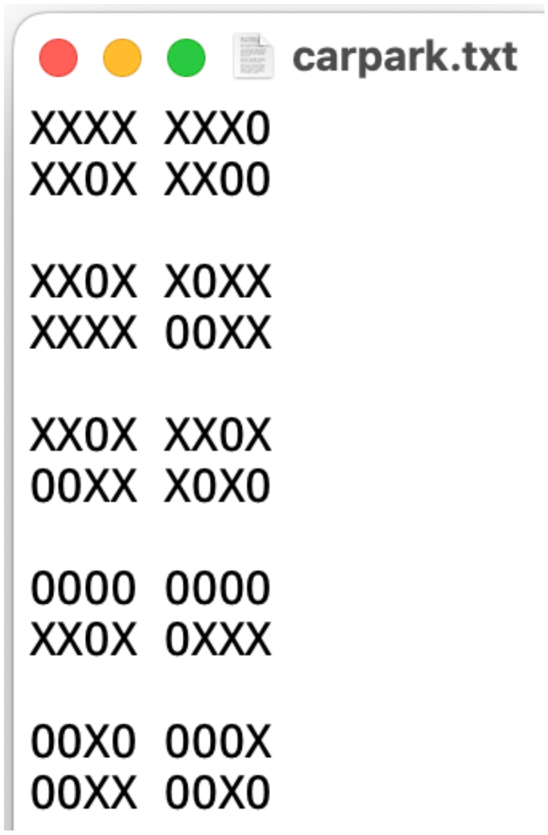
# Project I – SIE507 (20%)

*This project may be completed alone or in pairs (please let me know as soon as possible if you would like to work with a partner). The final submission should include an interactive python program. Submit all the source files in a .ZIP file. Please see below for the project brief and deliverables:* ***Due Nov 14th 11:59PM****. If you are working in pairs, only one member should submit the final materials to Brightspace (mention both names at the top of the source file as a comment).*

An automated car park (parking lot) is controlled using a sensor platform. The sensor platform updates **carpark.txt** with the latest status of the carpark. Text file **carpark.txt** contains the layout of the car park with 14 rows and 9 columns (which includes 4 empty rows and 1 empty column). Available slots are marked with the character ‘O’ (uppercase character O) and occupied slots are marked with character ‘X’.


carpark.txt

```
XXXX XXX0
XX0X XX00

XX0X X0XX
XXXX 00XX

XX0X XX0X
00XX X0X0

0000 0000
XX0X 0XXX

00X0 000X
00XX 00X0
```

Design and develop a Python program to remotely book a free slot in the car park. First, **the data should be read from the carpark.txt file into an appropriate python data structure (a list, tuple, or a dictionary).**

**Pro tip**: it will be helpful to prepare a flow chart of the program (considering all its functionalities and features) before you start coding. Remember – design first!

Possible error conditions should be appropriately handled with error messages and help texts.

**Getting started (sample skeleton of the code):**

Your program should implement the following functions.
You may alter the function parameters and define additional functions as required.

```python
def display(carpark_dataset):
    '''
    display the current layout of the car park (see screenshot for desired output)
    :param carpark_dataset: python data structure showing all the car park slots
    :return: None
    '''


def statistics(carpark_dataset):
    '''
    compute and display the percentage of occupancy %
    :param carpark_dataset: python data structure showing all the car park slots
    :return: None
    '''


def reserve(carpark_dataset):
    '''
    suggest any of the closest available slots from the building entrance (0, 0)
    if the user likes to reserve it, call the assign_slot() function to book it
    :param carpark_dataset: python data structure showing all the car park slots
    :return: True if successfully assigned, updated the data set and file; else False
    '''


def assign_slot(carpark_dataset, row, column):
    '''
    set an available slot as taken based on given row number and column letter
    update the data structure and file
    :param carpark_dataset: python data structure showing all the car park slots
    :param row: row to be assigned
    :param column: column to be assigned
    :return: True if successfully assigned, updated the data set and file; else False
    '''
```

**Tasks:**

1. **[20pts]** There should be a repeatable text-based menu with help text as below:

```
Select your choice from the menu:
D to display current status of the car park
R to reserve an empty slot in the car park
S to display % occupancy of the car park
Q to quit
:
```

*After the colon (:), a user can enter the menu choice character (D, R, S, or Q) and press enter.*

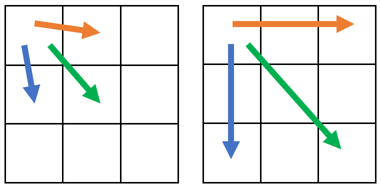
2. **[10pts]** The program should **quit** only if a user enters Q.

3. **[20pts]** The **display** option should: 1) read the data from the text file, and 2) display the data (display row numbers down the left side and column letters - A, B, C, D, E, F, G, and H - across the top line).

- There should be a blank column similar to the text file indicating the road but no blank rows

4. **[20pts]** The **reserve** function should suggest any of the closest available slots from the building entrance (assume that the building entrance is at the top left corner where row = 0 and column = 0).

a) Users can also provide a row number and column letter to reserve an empty slot.
b) If the data entered (row and column) is out of range, report an error message and the user can try again next pass of the menu loop.
c) If the seat is not available report an error message.
d) If the seat is available report so and update the data structure and text file (replace O with X).



To find the closest available slot, you should ***develop a nearest neighbor searching algorithm*** as illustrated.
(start searching for an empty slot in neighbors from the top left)

**Nearest Neighbor Searching**: develop an algorithm to search in terms of matrices, first, 2x2 (top left of the data set), then 3x3, followed by 4x4 and continue until 8x8. If no free slot can be found before 8x8 stop searching and ask for manual booking.

5. **[10pts]** The **statistics** function should report the % occupancy of the carpark. This is computed as the number of occupied slots divided by total slots. Set precision to 1 decimal.

***occupancy % = (number of occupied slots / total slots) * 100***

***NOTE: sample program execution and outputs are displayed at the end.***

**Special notes:**

**[10pts]** Comments are required in the following locations:

- At the top of the source code comment your name/s and a short program description.
- Comment the purpose of each variable.
- Comment major sections of code such as input, processing, output, functions, loops.

**[10pts]** Program Design: Your program should follow the listed specifications.

- Comments should be aligned and entered in a consistent fashion
- Blank lines should be added to aid readability (should be neat and easy to read)
- Comments should not contain spelling mistakes
- Variable names should be meaningful
- Define functions and data structures where necessary
- Error handling: handle possible error conditions and invalid user inputs

**<u>Sample program execution and outputs</u>**

```
Select your choice from the menu:
D to display current status of the carpark
R to reserve an empty slot in the carpark
S to display % occupancy of the carpark
Q to quit
: D

    ABCD  EFGH
1   XXXX  XXXO
2   XXOX  XXOO
3   XXOX  XOXX
4   XXXX  OOXX
5   XXOX  XXOX
6   OOXX  XOXO
7   OOOO  OOOO
8   XXOX  OXXX
9   OOXO  OOOX
10 OOXX  OOXO

Select your choice from the menu:
D to display current status of the carpark
R to reserve an empty slot in the carpark
S to display % occupancy of the carpark
Q to quit
: R

Closest available slot is 2C, do you want to reserve it (y/n)? n
Please enter your preferred empty slot: 3F
Your selection 3F is reserved!

Select your choice from the menu:
D to display current status of the carpark
R to reserve an empty slot in the carpark
S to display % occupancy of the carpark
Q to quit
: R

Closest available slot is 2C, do you want to reserve it (y/n)? y
Your selection 2C is reserved!
```

```
Select your choice from the menu:
D to display current status of the carpark
R to reserve an empty slot in the carpark
S to display % occupancy of the carpark
Q to quit
: S

Current occupancy is XX.X%

Select your choice from the menu:
D to display current status of the carpark
R to reserve an empty slot in the carpark
S to display % occupancy of the carpark
Q to quit
: D

   ABCD  EFGH
1  XXXX  XXXO
2  XXXX  XXOO
3  XXOX  XXXX
4  XXXX  OOXX
5  XXOX  XXOX
6  OOXX  XOXO
7  OOOO  OOOO
8  XXOX  OXXX
9  OOXO  OOOX
10 OOXX  OOXO


Select your choice from the menu:
D to display current status of the carpark
R to reserve an empty slot in the carpark
S to display % occupancy of the carpark
Q to quit
:
```

* Note that the 3F and 2C slots are now occupied in the dataset. To achieve this, you need to write the updated dataset to the carpark.txt file every time you update the dataset. Please do not change the format of the dataset in carpark.txt when you append data.