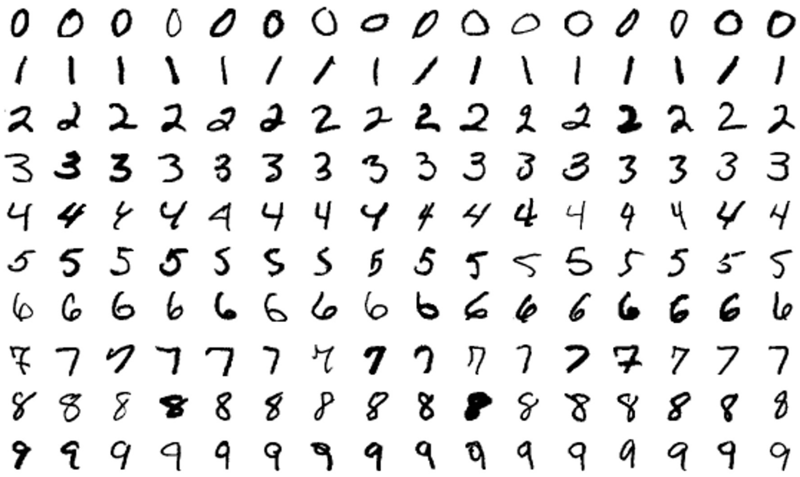
Assessed Coursework 2

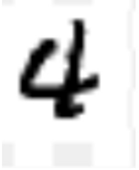
Deadline: Friday, Week 12.

Task: Given a data set (MNIST) containing images of handwritten digits, implement a simple classification algorithm (k-nearest neighbor), which labels a test image with "1", or "not 1"

Dataset:



Input:



Output:

“4”

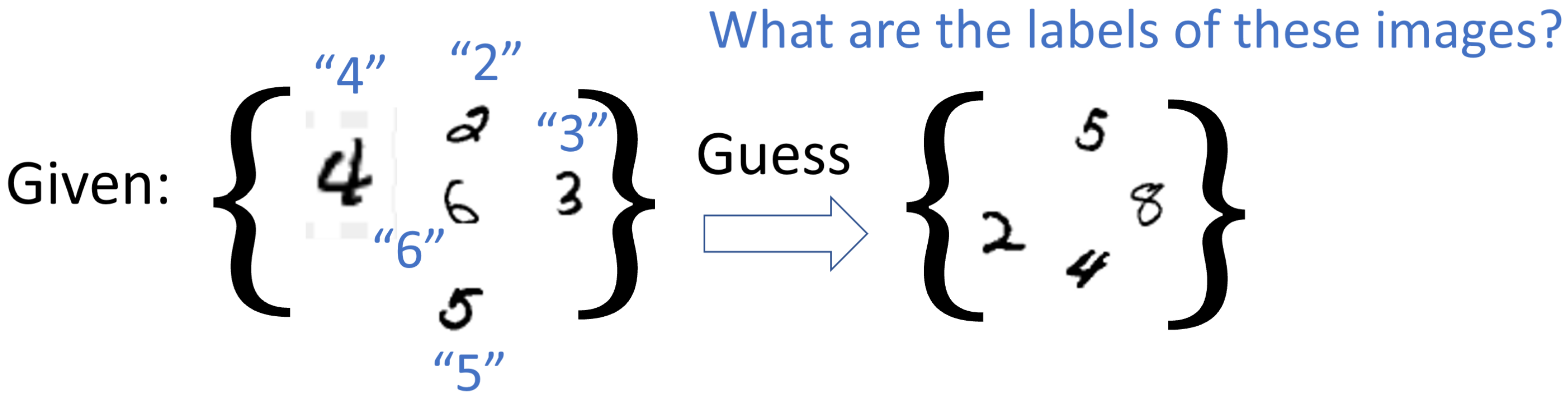
Input:

Output:

“1”

Assessed Coursework 2

- In this CW, the goal is to use images and labels in one dataset to predict labels of images in another dataset, where labels are NOT observed.



What are the labels of these images?

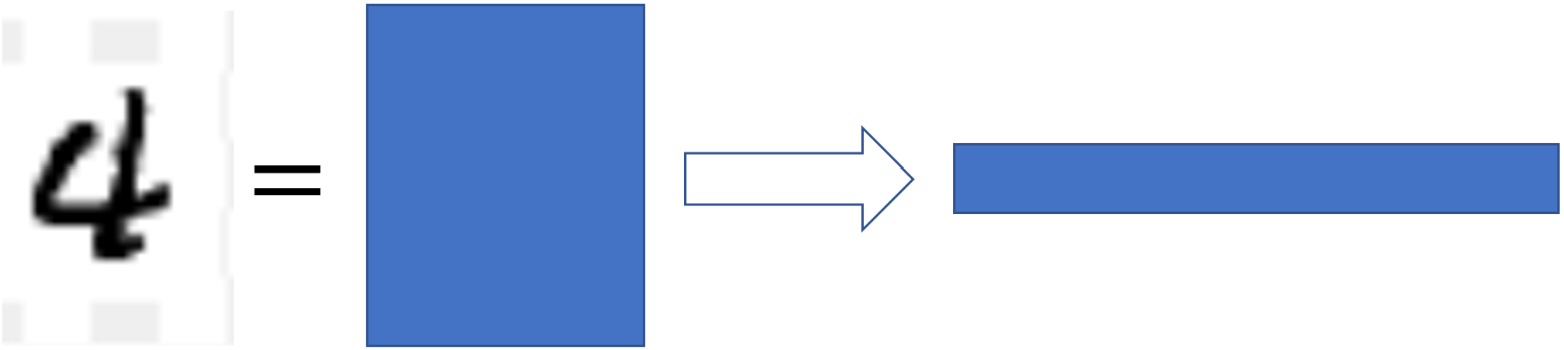
Given:

“4” “2” “3” “6” “5”

Guess

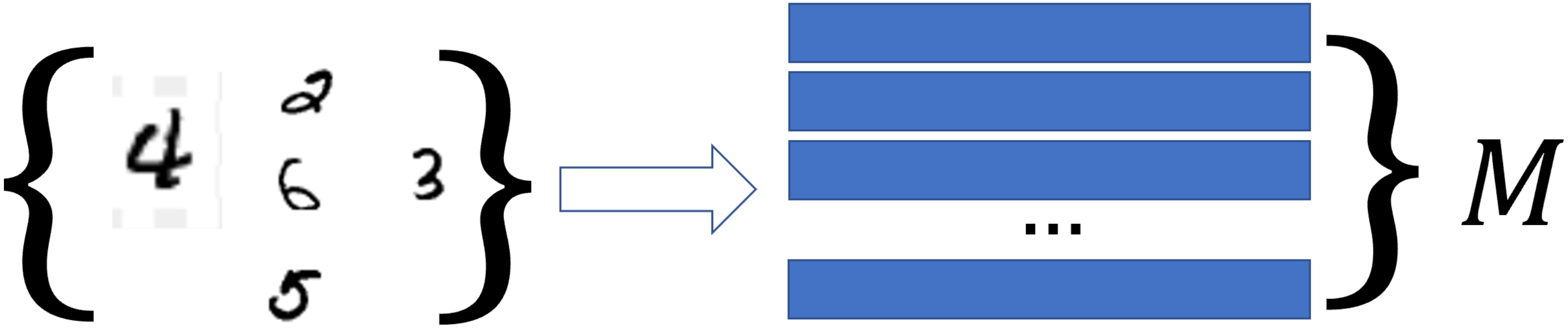
Assessed Coursework 2

- As we mentioned in previous lectures/labs, images are stored as flattened matrices (in row/col major order) in the memory.
- Each image is stored as a vector in this coursework.



Assessed Coursework 2

- In this CW, we are dealing with sets of images. Stacking all the image vectors together, you get a matrix.

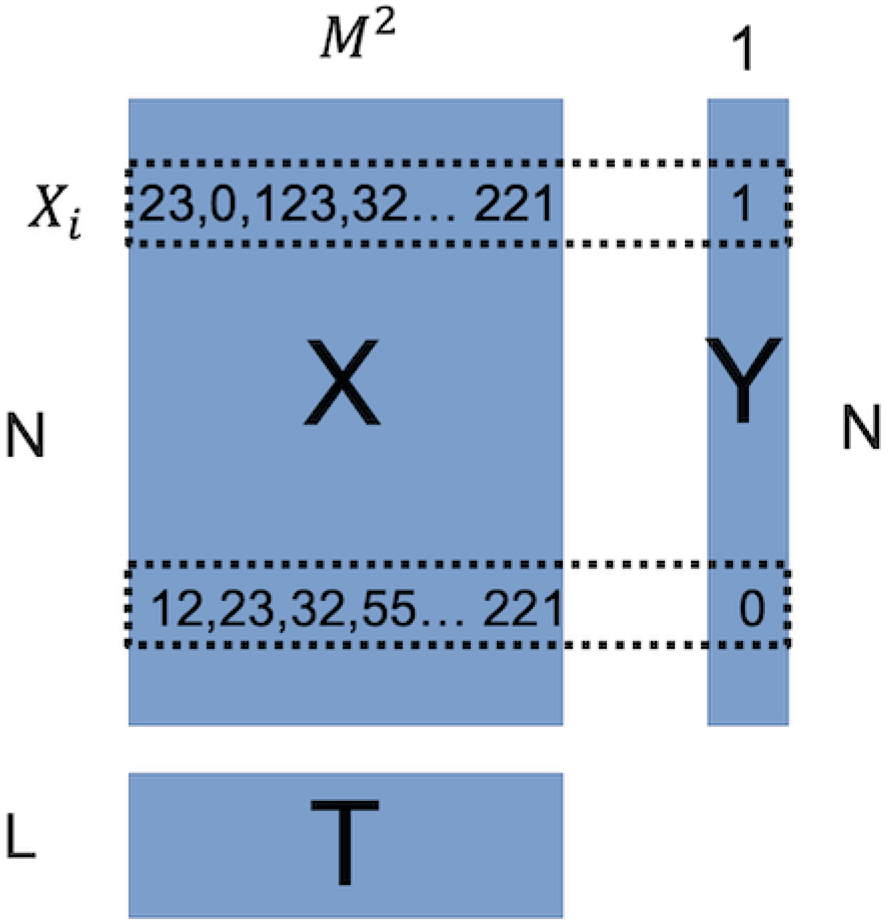


- Specifically, there are two sets of images in this CW, so they are represented by two matrices X, T.

Part I, The Data Set

- CW folder contains 3 `.matrix` files storing 3 matrices.
  - `X.matrix` stores an N by M^2 matrix X where each row is a grayscale M by M image stored in row-major order.
  - `T.matrix` stores an L by M^2 matrix T where each row is an M by M **test image** in row major order.
  - `Y.matrix` stores an N by 1 matrix Y where each row is a scalar, indicating whether the corresponding row in X is digit 1 or not.
  - X and Y together are called "training set" in machine learning, while T is the "testing set". Y is called the "labels" of X.

Part I, Data Structure



- If $Y_i = 1$, then the image X_i is a handwritten digit 1. If $Y_i \neq 1$, the image X_i is NOT a handwritten digit 1.

Part I, Loading Dataset

- The code for loading these images from files have been provided to you. Matrices are represented by a **matrix structure** in this coursework.

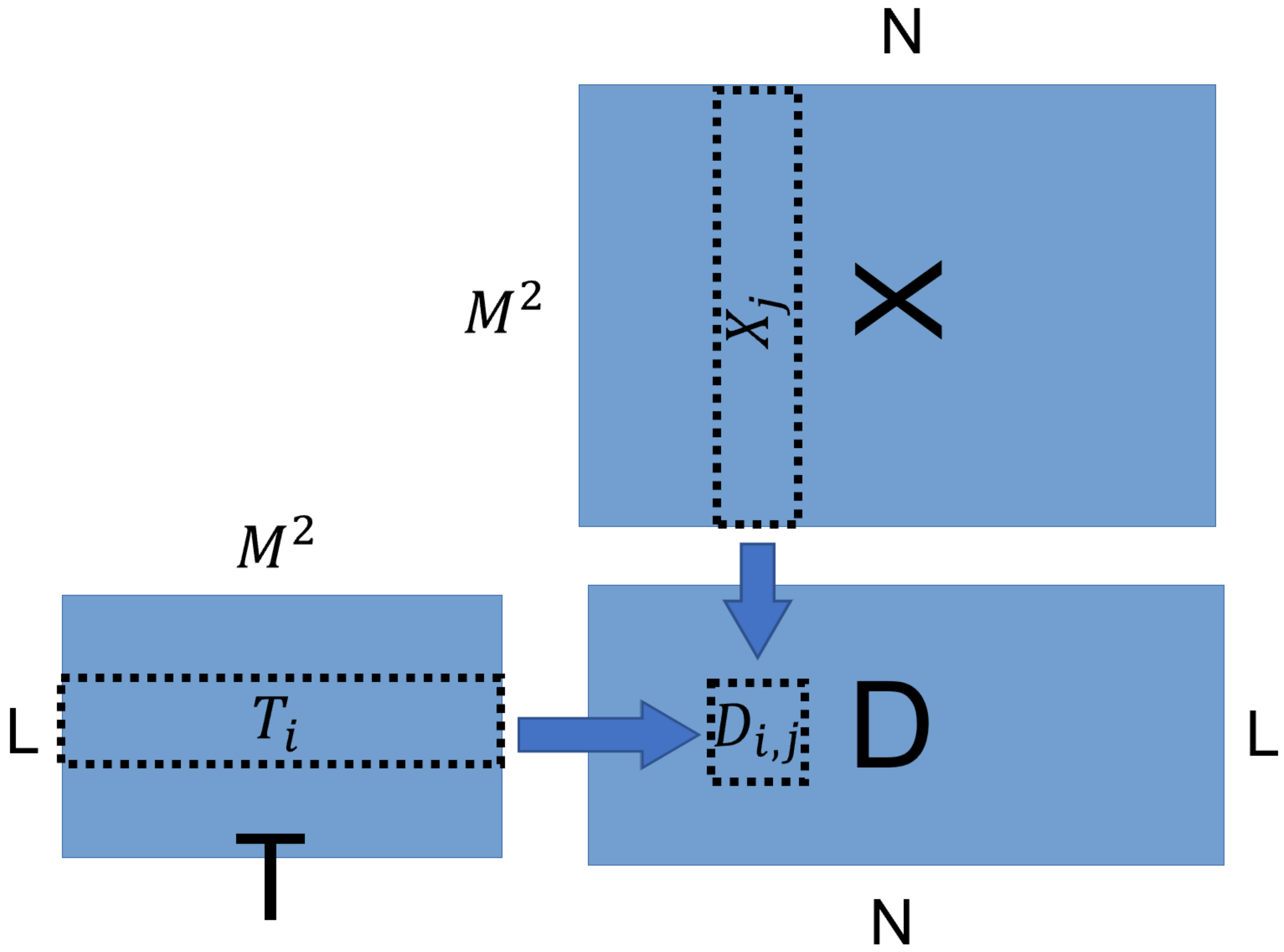
```c
struct matrix
{
    int numrow; //number of rows
    int numcol; //number of columns
    int *elements; // pointer pointing to an integer array
    // storing all entries in the matrix in row major order.
};
typedef struct matrix Matrix;
// now "Matrix" is an alias of "struct matrix"
```

- By simply running the skeleton code, you should see some basic statistics of X, Y and T .
    - What are M, N and L?
    - How many images in the training set X are digit 1?

Part II Computing Distance Matrix D (40pt in total)

- Construct an L by N matrix D, where the i, j-th element $D_{ij} = \mathrm{dist2}(T_i, X_j)$
    - T_i is the i-th row of T.
    - X_j is the j-th row of X
- $\mathrm{dist2}(a, b)$ computes the squared euclidean distance between two vectors a and b with K elements.
    - $\mathrm{dist2}(a, b) := \sum_{k=1}^{K} (a_k - b_k)^2$.

Part II (Computing D)



Part II.1 (15pt) Constructing D

Before your `main` function,

1. Write a few helper functions:

-
  ```
  int get_elem(Matrix M, int i, int j)
  ```

  returns the `i, j` th element of matrix `M`.

-
  ```
  void set_elem(Matrix M, int i, int j, int value)
  ```

  assign `value` to the `i, j` th element of matrix `M`
  In this coursework, `i, j` are **zero-based indices**.

In your `main` function,

2. Allocate HEAP memory for D.
3. Declare and initialize a new `matrix` variable `D`.

Part II.2 Computing D (25pt)

Now, populate the matrix D with correct values.

- **Hint**: Compare the computation of D and the matrix multiplication. What are the similarities and what are the dissimilarities?
    - Can you modify the matrix multiplication code to compute matrix D?
- **Hint**, you can write a function

  ```
  void pairwise_dist(Matrix T, Matrix X, Matrix D)
  ```

    - where D is the output, storing the outcome.
- Partial points will be given for correctly written code for computing $\mathrm{dist2}(a, b)$.

Part III.1 Guessing Labels (15%)

- For each row of matrix D, find **the indices** of the five smallest elements.
    - Suppose the i-th row of D is $[3, 2, 5, 1, 2, 5, 13, 46, 32]$,
    - The indices of the five smallest elements are $[3, 1, 0, 5, 6]$.

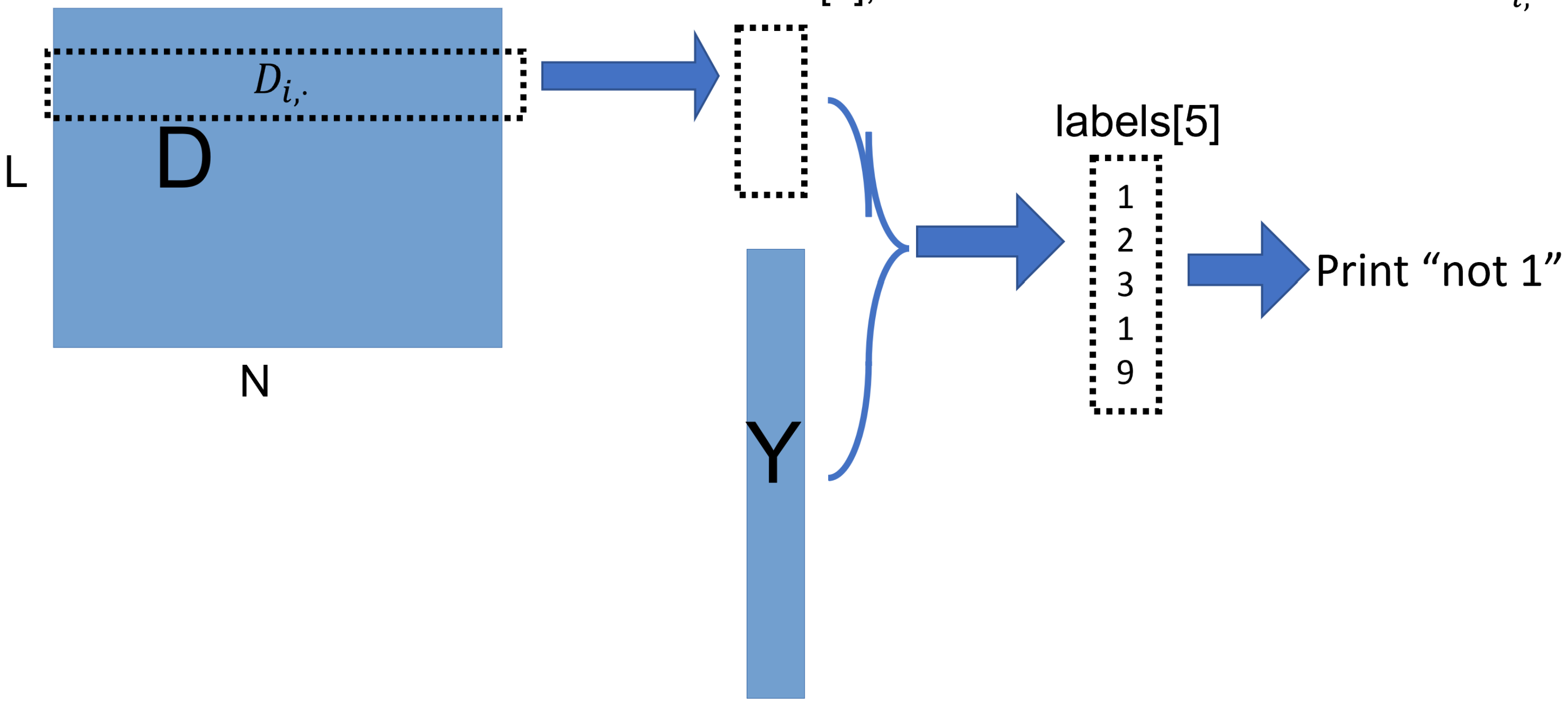
Part III.2 Guessing Labels (15%)

- Now, suppose the array `indices` contains the indices of the five smallest elements in row i of matrix D.
  - Create a new array `labels` with length 5.
  - Assign the value of $Y_{\text{indices}[k],0}$ to the k-th element in `labels`.
  - Count the number of `1` in `labels`.
  - If the count >= 3, print out `1`. Otherwise, print `not 1`.
- Repeat above for all rows in D.

Part III Guessing Labels (30pts total)

For each row in D, do:

Indices[5], indices of 5 smallest elements in $D_{i,\cdot}$



labels[5]

1
2
3
1
9

Print “not 1”

Part III Guessing Labels

- At each row D_i, the print-out is your "guess" of the testing image T_i using 5-nearest neighbour algorithm.
    - If the print-out is `1`, it means the algorithm thinks the image T_i is a digit 1.
    - If the print-out is `not 1`, it means the algorithm thinks the image T_i is NOT a digit 1.
- After your guess, you can optionally print the image stored in T_i to the console to validate your guess.

Part III Guessing Labels

- Hint: Write a helper function

  ```
  void minimum5(int len, int a[], int indices[])
  ```

  It takes an array `a` with length `len` as input, then fills `indices[]` with the indices of the five smallest elements.

- You might want to test your functions in a separate c file to ensure that they are correctly written.

Final Project: Marking Criteria

- Submitting correct code (10%)
    - Submitting a C file with **the correct name**.
    - Your code compiles and runs **without major error** such as **crash**, **infinite loop**.
        - It will be tested using `gcc` in the lab pack.
- Part II 40% (15% + 25%)
- Part III 30% (15% + 15%)
- Good Coding Practice (20%)
    - Good code format
    - Good variable naming scheme.
    - Apt comments

Final Project: Dos and Don'ts

- You can discuss with your classmates about general strategies but write your own code!
- Don't give your code to other students.
- Review relevant previous lab sessions before you start.
- You can use whatever material you can find to help you complete the task, but you need to add a reference in the comments.
- You are only allowed to use standard features of C.
    - You can use `stdio.h`, `stdlib.h`, `limits.h` and `math.h`.
    - If you want to use other libraries, consult with the lecturer or TA beforehand.

Final Project: Q&A

- We will answer questions posted on the Blackboard forum or answering them during the lab sessions.
- We will inspect the forum regularly and try to respond in 24 hours.