1 Introduction

- This is a practical exercise that must be done in the lab during the practical session.
- This practical must be completed in pairs.
- The practical consists of two algorithmic challenges.

2 Constraints

1. You may create any necessary functions that you require.
2. You must complete this assignment with your partner.
3. You may ask the Teaching Assistants for help but they will not be allowed to give you the solutions.
4. You are free to utilise any online resources to your disposal.
5. You have to submit your solution before the end of your booked time slot.

3 Submission Instructions

You are required to upload all your source files as a single tar.gz or .zip archive to the CS website before the deadline. Clearly separate Task 1 and Task 2 by placing them in separate folders. Your student numbers must be clearly reflected in all source files as a comment at the top of the file.

Both members of the pair are required to submit and upload all source files in the format as stated above.

4 Assessment

- You will be assessed on the correctness of your solution.
- You will be assessed on how you work together.
- Marks will by default contribute equally to the COS121 practical mark of both partners. However, this may be different if one or both are not COS121 students, or if there is a good reason for not allocating equal marks to both partners. Note that this is a Pair Programming Exercise, therefore, if one individual obviously works more in the process of solving the problem, both partners will get extremely low marks because they failed to do pair programming.
5 Assignment Instructions

This practical assignment consists of two independent tasks. Both tasks need to be completed in order to obtain full marks.

**Task 1: Matrix Multiplication** (20 marks)

Your task is to implement a program which will compute the dot product of two matrices. Your solution should comply with the following requirements:

1. The user should be asked for the dimensions (number of columns and rows) of matrix A.
2. The user should be asked for the dimensions (number of columns and rows) of matrix B.
3. None of the dimensions are allowed to be larger than 10 nor less than 1. An error should be displayed if this is the case and prompt the user for new dimensions until they are valid.
4. The user should be asked for the individual values of the matrix A.
5. The user should be asked for the individual values of the matrix B.
6. Compute the dot product of the matrix A and B as matrix C.
7. Output matrix C in an appropriate manner.

See the screenshot below as an example of a working program:

![Screenshot](https://example.com/screenshot.png)

Your program will be tested with the following two matrix equations:

\[
C = \begin{bmatrix}
0 & 4 & 6 \\
3 & 2 & 5 \\
7 & 8 & -3
\end{bmatrix}
\begin{bmatrix}
-2 & 4 & 1 & 6 \\
3 & 9 & 5 & -1 \\
2 & -4 & 0 & 7
\end{bmatrix}
\]

\[
C' = \begin{bmatrix}
1 & 3 & 5 \\
2 & 4 & 6
\end{bmatrix}
\begin{bmatrix}
3 & 6 \\
1 & 4 \\
5 & 2
\end{bmatrix}
\]

Other unseen matrices may also be tested.

Task 2: Ackermann .............................................................. (20 marks)

In Computer Science it is a very common to use recursive algorithms. So also in Mathematics. The Ackermann sequence is an example of such a recursive function. The Ackermann function can be mathematically defined as:

\[
A(m, n) = \begin{cases} 
  n + 1 & \text{if } m = 0 \\
  A(m-1, 1) & \text{if } m > 0 \text{ and } n = 0 \\
  A(m-1, A(m, n-1)) & \text{if } m > 0 \text{ and } n > 0 
\end{cases}
\]  

(1)

If this is to be translated to C++ the function would be as follows:

```cpp
unsigned int ackermann(unsigned int m, unsigned int n) {
  if (m == 0) {
    return n + 1;
  }
  if (n == 0) {
    return ackermann(m - 1, 1);
  }
  return ackermann(m - 1, ackermann(m, n - 1));
}
```

Your task is to re-implement the Ackermann function as a non-recursive (i.e. iterative) version. Check your implementation against the values returned for m in the range 0 .. 3 and n in the range 0 .. 4. Output the computed values to the terminal.

TIP: There is a data structure called a stack which you may utilise if you so wish. Simply add the C++ std stack library to your program as a dependency. For details about using a stack refer to the following web sites:

http://www.cplusplus.com/reference/stack/stack/
http://www.cplusplus.com/reference/stack/stack/stack/