Recap

- Defining and Manipulating 1D Arrays
- Representing 2D arrays as 1D arrays
- Today
  - Multi-Dimensional Arrays
  - Matrices
  - Operations of Matrices
  - The Matrix Package
Two-dimensional Arrays

- Two indices needed to reference elements in the array

<table>
<thead>
<tr>
<th>City</th>
<th>Amsterdam</th>
<th>Berlin</th>
<th>London</th>
<th>Madrid</th>
<th>Paris</th>
<th>Rome</th>
<th>Stockholm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amsterdam</td>
<td>0</td>
<td>648</td>
<td>494</td>
<td>1752</td>
<td>495</td>
<td>1735</td>
<td>1417</td>
</tr>
<tr>
<td>Berlin</td>
<td>648</td>
<td>0</td>
<td>1101</td>
<td>2349</td>
<td>1092</td>
<td>1588</td>
<td>1032</td>
</tr>
<tr>
<td>London</td>
<td>494</td>
<td>1101</td>
<td>0</td>
<td>1661</td>
<td>404</td>
<td>1870</td>
<td>1807</td>
</tr>
<tr>
<td>Madrid</td>
<td>1752</td>
<td>2349</td>
<td>1661</td>
<td>0</td>
<td>1257</td>
<td>2001</td>
<td>3138</td>
</tr>
<tr>
<td>Paris</td>
<td>495</td>
<td>1092</td>
<td>404</td>
<td>1257</td>
<td>0</td>
<td>1466</td>
<td>1881</td>
</tr>
<tr>
<td>Rome</td>
<td>1735</td>
<td>1588</td>
<td>1870</td>
<td>2001</td>
<td>1466</td>
<td>0</td>
<td>2620</td>
</tr>
<tr>
<td>Stockholm</td>
<td>1417</td>
<td>1032</td>
<td>1807</td>
<td>3138</td>
<td>1881</td>
<td>2620</td>
<td>0</td>
</tr>
</tbody>
</table>

-- various constants used in data types
max_dist : constant := 40077; -- max distance on earth

-- type declarations

type Distances is range 0 .. max_dist;
type City is (Amsterdam, Berlin, London, Madrid, Paris, Rome, Stockholm);
type distance_table is array (City, City) of Distances;

-- distances between various European cities
inter_city : distance_table :=
  -- Amst, Berl, Lond, Madr, Pari, Rome, Stock
  (( 0, 648, 494, 1752, 495, 1735, 1417), -- Amsterdam
   ( 648, 0, 1101, 2349, 1092, 1588, 1032), -- Berlin
   ( 494, 1101, 0, 1661, 404, 1870, 1807), -- London
   (1752, 2349, 1661, 0, 1257, 2001, 3138), -- Madrid
   ( 495, 1092, 404, 1257, 0, 1466, 1881), -- Paris
   (1735, 1588, 1870, 2001, 1466, 0, 2620), -- Rome
   (1417, 1032, 1807, 3138, 1881, 2620, 0)); -- Stockholm

-- distances I have traveled between various cities
traveled : distance_table := (others => (others => 0));
your_travel : distance_table;
Using 2-D Arrays

- To reference elements of a 2D array variable, use both index values
  \[
  \text{put(inter\_city(Berlin, Rome));}
  \text{traveled(Stockholm, London) := 1807;}
  \]

- Nested loops are often used to process 2D arrays
  -- write out the table
  \[
  \text{for from in Amsterdam .. Stockholm loop}
  \quad \text{-- write one line of the table}
  \text{for to in Amsterdam .. Stockholm loop}
  \quad \text{PUT(inter\_city(from, to), width=>6);}
  \quad \text{end loop;}
  \text{NEW\_LINE;}
  \text{end loop;}
  \]

Multi-dimensional arrays

- Often have information in a tabular form
  - Tables of data
  - Matrices
- Use a multi-dimensional array to repr. data
  - Items indexed by several subscripts
  - E.g., row and column for 2D arrays
- Can have as many dimensions as wanted
  - Extend declaration to include required index ranges
  - Extend references to include required indices
Multi-dimensional Array

- **type** multidim is
  - array \( (\text{range}_1, \text{range}_2, \ldots, \text{range}_n) \)
  - of element-type;

- **Example:**
  - **type** YearByMonth is array \((1900..1999, \text{Month})\) of real;
  - **type** Election is array \((\text{Candidate}, \text{precinct})\) of integer;

  -- type declaration for higher dimensional arrays
  **type** CUBE6 is array \((1..6, 1..6, 1..6)\) of CHARACTER;

  -- variable declaration for higher dimensional arrays
  **tictactoe_3d** : CUBE6;

  -- reference to element in multi-dimensional array
  **PUT**(tictactoe_3d(2,3,4));

---

**Concept Question - 1**

What are the dimensions of the Array A
1. 3,3,3
2. 2,3,2
3. Don’t Know
4. It is dimensionless
Concept Question -2

What is the value of N displayed?
1. 12
2. 0
3. Will throw a constraint error
4. Don’t Know

Basics

• Scalar – is a number, represented as [a] or [1]
• Vector – is a single row or column of numbers, denoted by a small bold letter
  – Row vector [ 1 2 3 4 5]
  – Column vector

\[
\begin{bmatrix}
1 \\
2 \\
3 \\
4 \\
5
\end{bmatrix}
\]
Matrix

A matrix is a set of rows and columns of numbers – denoted by a **bold Capital letter**

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

The **Order** of a matrix is the number of rows x number of columns in the matrix

\[
A = \begin{bmatrix}
1 & 2 & 3 \\
4 & 5 & 6
\end{bmatrix}_{(2x3)}
\]

Operations

- Equality
- Addition/Subtraction
- Multiplication
- Determinant
- Inversion
Matrix Equality

- Two matrices are said to be equal iff they have the same order and all the elements are equal.

\[
\begin{bmatrix}
a_{11} & a_{12} \\
a_{21} & a_{22} \\
a_{31} & a_{32}
\end{bmatrix}
+ \begin{bmatrix}
b_{11} & b_{12} \\
b_{21} & b_{22} \\
b_{31} & b_{32}
\end{bmatrix}
= \begin{bmatrix}
c_{11} & c_{12} \\
c_{21} & c_{22} \\
c_{31} & c_{32}
\end{bmatrix}
\]

Matrix Addition

- Two matrices A, B can be added iff they have the same order.

- The resulting matrix has the same order and the elements in the new matrix are defined as \( \forall i, j, c_{ij} = a_{ij} + b_{ij} \)
Matrix Multiplication

- Scalar multiplication – multiply each element in the matrix by the scalar
- To multiply two matrices, they must be conformable (number of rows of the 1st matrix = number of columns in the 2nd matrix)
- When can you multiply two matrices $A_{mxn}, B_{pxq}$?

Matrix Multiplication

- Consider two matrices $A_{mxn}, B_{pxq}$
- $C_{mxq} = AB$

\[
\begin{bmatrix}
    a_{i1} & a_{i2} & a_{i3} \\
    a_{21} & a_{22} & a_{23} \\
    a_{31} & a_{32} & a_{33}
\end{bmatrix} \times \begin{bmatrix}
    b_{11} & b_{12} \\
    b_{21} & b_{22} \\
    b_{31} & b_{32}
\end{bmatrix} \Rightarrow \begin{bmatrix}
    c_{11} & c_{12} \\
    c_{21} & c_{22} \\
    c_{31} & c_{32}
\end{bmatrix}
\]

\[
C(i,j) = \sum_{k=1}^{n} A(i,k) \times B(k,j)
\]

i ranges from 1..m
j ranges from 1..q
k ranges from 1..n = p
Matrix Transpose

- A transposed matrix has the elements in the rows and columns interchanged.
- The transpose of A is represented as $A'$

$$A = \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix}, \quad A' = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \quad \forall \ A'(i,j) := A(j,i)$$

Matrix Determinant

- The determinant of a matrix A is denoted by $|A|$ or det $A$.
- Determinants exist only for square matrices

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}, \quad |A| = a_{11}a_{22} - a_{12}a_{21}$$
**Generic Determinant**

- For any nxn matrix, the formula for finding the determinant is

\[
|A| = \sum_{j=1}^{n} s_j a_{1j} \det A_j
\]

- \(s_j\) is +1 if \(j\) is odd and -1 if \(j\) is even
- \(a_{1j}\) is the element in row 1 and column \(j\)
- \(A_j\) is the n-1 x n-1 matrix obtained from matrix A by deleting its row 1 and column \(j\) (cofactor matrix).

**3x3 Determinant**

- If A is a 3x3 matrix shown below,

\[
\begin{bmatrix}
a_{11} & a_{12} & a_{13} \\
a_{21} & a_{22} & a_{23} \\
a_{31} & a_{32} & a_{33}
\end{bmatrix}
\]

- The determinant \(|A|\) is given by

\[
|A| = a_{11}a_{22}a_{33} - a_{11}a_{23}a_{32} + a_{12}a_{23}a_{31} - a_{12}a_{21}a_{33} + a_{13}a_{21}a_{32} - a_{13}a_{22}a_{31}
\]
**Adjoint Matrix**

If $C_{ij}$ is the cofactor of $a_{ij}$, then $\text{Adj } \mathbf{A} = [C_{ji}] = [C_{ij}]^T$.

\[
\begin{bmatrix}
  1 & 0 & -2 \\
  2 & 3 & 0 \\
  1 & 2 & -1
\end{bmatrix}
\]

then the matrix of cofactors of $\mathbf{A}$ is:

\[
\begin{bmatrix}
  3 & 0 & 2 & 0 & 2 & 3 \\
  -2 & -1 & 1 & -1 & 1 & 2 \\
  0 & -2 & 1 & -2 & -1 & 0 \\
  -2 & -1 & 1 & -1 & 1 & 2 \\
  0 & -2 & 1 & -2 & 1 & 0 \\
  3 & 0 & 2 & 0 & 2 & 3
\end{bmatrix}
\]

\[
\text{Adj} (\mathbf{A}) = \begin{bmatrix}
  -3 & -4 & 6 \\
  2 & 1 & -4 \\
  1 & -2 & 3
\end{bmatrix}
\]

i.e. the transpose of the above

**Inversion**

- A matrix is *singular* if it does not have an inverse (the determinant is 0).
- The formula for finding the inverted matrix is given as:

\[
\mathbf{A}^{-1} = \frac{\text{Adj} \mathbf{A}}{|\mathbf{A}|} \quad (|\mathbf{A}| \neq 0)
\]
Ada95 Matrix Package

- [http://dflwww.ece.drexel.edu/research/ada/](http://dflwww.ece.drexel.edu/research/ada/)
- The archive of this matrix package is available in tar or zip format
- Link available from CP web page, today's lecture

The Matrix Package

- **package** Generic_Real_Arrays: basic math functions and array math routines as defined by the Ada 95 ISO document referred to above for vectors and matrices of real numbers.
- **package** Generic_Real_Arrays.Array_IO: routines to print vectors and arrays of real numbers to the console.
- **package** Generic_Real_Arrays.Operations: more advanced functions for vectors and arrays of real numbers, including dynamic allocation, subvectors and submatrices, determinants, eigenvalues/vectors, singular value decomposition, and inverses.
- **package** Real_Arrays_Operations_Test: test program demonstrating the use of every subprogram in Generic_Real_Arrays.Operations via a functional test.