Cairo University
Faculty of Computers and Artificial Intelligence
Computer Science Department

Programming-1 CS112 **Application: Matrix multiplication** Dr. Amin Allam

1 Matrix multiplication (Static)

Suppose that a business man is interested in the sales of 4 devices (phones, cameras, keyboards, monitors) during each of the last 3 months.

```
1  enum{PHONE,CAMERA,KEYBOARD,MONITOR};
2  int a[3][4];          // a[i][j]=v means he sold v of device j in month i
3  a[0][PHONE]=5;        // He sold 5 phones in the first month
4  a[0][CAMERA]=7;       // He sold 7 cameras in the first month
5  a[0][KEYBOARD]=4;     // He sold 4 keyboards in the first month
6  a[0][MONITOR]=8;      // He sold 8 monitors in the first month
7  a[1][PHONE]=6;        // He sold 6 phones in the second month
8  a[1][CAMERA]=2;       // He sold 2 cameras in the second month
9  a[1][KEYBOARD]=3;     // He sold 3 keyboards in the second month
10 a[1][MONITOR]=9;      // He sold 9 monitors in the second month
11 a[2][PHONE]=1;        // He sold 1 phone in the third month
12 a[2][CAMERA]=4;       // He sold 4 cameras in the third month
13 a[2][KEYBOARD]=6;     // He sold 6 keyboards in the third month
14 a[2][MONITOR]=2;      // He sold 2 monitors in the third month
```

The values of this matrix are as follows:

a[0][0]=5	a[0][1]=7	a[0][2]=4	a[0][3]=8
a[1][0]=6	a[1][1]=2	a[1][2]=3	a[1][3]=9
a[2][0]=1	a[2][1]=4	a[2][2]=6	a[2][3]=2

Suppose that the buying and selling prices of these 4 devices are organized in a matrix as follows:

```
1  double b[4][2];        // b[i][0] is the buying price of device i
2                         // b[i][1] is the selling price of device i
3  b[PHONE][0]=70;        // The buying price of phone is 70
4  b[PHONE][1]=75;        // The selling price of phone is 75
5  b[CAMERA][0]=50;       // The buying price of camera is 50
6  b[CAMERA][1]=53;       // The selling price of camera is 53
7  b[KEYBOARD][0]=6;      // The buying price of keyboard is 6
8  b[KEYBOARD][1]=7.5;    // The selling price of keyboard is 7.5
9  b[MONITOR][0]=23;      // The buying price of monitor is 23
10 b[MONITOR][1]=26;      // The selling price of monitor is 26
```

The values of this matrix are as follows:

b[0][0]=70	b[0][1]=75
b[1][0]=50	b[1][1]=53
b[2][0]=6	b[2][1]=7.5
b[3][0]=23	b[3][1]=26

Suppose that the business man wants to know the total buying and selling prices of all devices he sold during each of the last 3 months, he can get this information by multiplying the two previous matrices as follows:

```
1  void MatrixMultiply(int a[3][4],double b[4][2],double w[3][2])
2  {
3      // ra and ca are number of rows and columns of matrix a (input)
4      // rb and cb are number of rows and columns of matrix b (input)
5      // rw and cw are number of rows and columns of matrix w (output)
6      // rb=ca, rw=ra, cw=cb
7
8      int ra=3, ca=4, cb=2;
9      int i, j, k;
10     for(i=0;i<ra;i++) for(j=0;j<cb;j++) // Compute w[i][j]
11     {
12         w[i][j]=0;                      // Initialize w[i][j]
13         for(k=0;k<ca;k++)
14             w[i][j]+=a[i][k]*b[k][j];
15     }
16 }
```

The values of the matrix resulting from multiplying `a` and `b` are:

w[0][0]=908	w[0][1]=984
w[1][0]=745	w[1][1]=812.5
w[2][0]=352	w[2][1]=384

where `w[i][0]` is the total buying price of all devices sold in month `i` and `w[i][1]` is the total selling price of all devices sold in month `i`.

$\Rightarrow$ What is the result of multiplying $\begin{bmatrix}1 & 1 & 1\end{bmatrix}$ by the matrix `w`? What do the values mean?

$\Rightarrow$ What is the result of multiplying `w` by $\begin{bmatrix}1\\-1\end{bmatrix}$? What do the values mean?

2 Matrix multiplication (Dynamic)

Suppose that we want to implement a general program that takes two arbitrary matrices and multiply them. The user will provide the matrices dimensions as well as matrices values. We should allocate matrices dynamically and use pointers as follows:

```
1 double** AllocateMatrix(int r, int c) // r=num rows, c=num cols
2 {
3    double** a=new double*[r];
4    for(int i=0;i<r;i++) a[i]=new double[c];
5    return a;
6 }
```

Matrices should be dynamically deallocated as follows:

```
1 void DeallocateMatrix(double** a, int r) // r=num rows
2 {
3    for(int i=0;i<r;i++) delete[] a[i];
4    delete[] a;
5 }
```

Matrix values can be obtained from the user as follows:

```
1 void InputMatrix(double** a, int r, int c)
2 {
3    for(int i=0;i<r;i++) for(int j=0;j<c;j++) cin>>a[i][j];
4 }
```

Matrix values can be output to the user as follows:

```
1 void OutputMatrix(double** a, int r, int c)
2 {
3    for(int i=0;i<r;i++)
4    {
5       for(int j=0;j<c;j++) cout<<a[i][j]<<" ";
6       cout<<endl;
7    }
8 }
```

Matrix multiplication can be done as follows:

```
void MatrixMultiply(double** a, double** b, double** w,
                    int ra, int ca, int cb)
{
   // ra and ca are number of rows and columns of matrix a (input)
   // rb and cb are number of rows and columns of matrix b (input)
   // rw and cw are number of rows and columns of matrix w (output)
   // rb=ca, rw=ra, cw=cb

   int i, j, k;
   for(i=0;i<ra;i++) for(j=0;j<cb;j++) // Compute w[i][j]
   {
      w[i][j]=0;                      // Initialize w[i][j]
      for(k=0;k<ca;k++)
         w[i][j]+=a[i][k]*b[k][j];
   }
}
```

A complete program can be written as follows:

```
int main()
{
   int ra, ca, cb;
   cin>>ra>>ca>>cb;

   int rb=ca, rw=ra, cw=cb;

   double** a=AllocateMatrix(ra, ca);
   double** b=AllocateMatrix(rb, cb);
   double** w=AllocateMatrix(rw, cw);

   InputMatrix(a, ra, ca);
   InputMatrix(b, rb, cb);

   MatrixMultiply(a, b, w, ra, ca, cb);

   OutputMatrix(w, rw, cw);

   DeallocateMatrix(a, ra);
   DeallocateMatrix(b, rb);
   DeallocateMatrix(w, rw);

   return 0;
}
```