Name and, if possible, ID#:___\

AMERICAN UNIVERSITY OF ARMENIA
College of Science and Engineering
COMP120 Introduction to Object-Oriented Programming
MIDTERM 2 EXAM

Date: Tuesday, March 24 2015
Starting time: 10:30
Duration: 1 hour 20 minutes
Attention: **ANY COMMUNICATION IS STRICTLY PROHIBITED**
Please write down your name at the top of all used pages

5/15

Problem 1

The easiest way to implement rotation by 90^0 of a square array is to transpose it and then reverse all its rows separately. Transposing once more after the rotation will result in vertical flip – the top row will appear at the bottom, the second row will become the last but one, etc. Write a C++ function ***void flip(int *a2D, int size)*** that takes as its argument a pointer to the first element of a square array ***int *a2D*** of the specified ***int size*** and flips it vertically. Use already implemented functions ***void reverse(int a1D[], int length)*** and ***void transpose(int *a2D, int size)***:

```
void reverse(int a1D[], int length)
{
      for (int i = 0; i < length / 2; i++)
            swap(a1D[i], a1D[length - 1 - i]);
}

void transpose(int *a2D, int size)
{
      for (int row = 0; row < size; row++)
            for (int col = row + 1; col < size; col++)
                  swap(a2D[row * size + col], a2D[col * size + row]);
}
```

```
Void flip (int *a2D, int size)
{  transpose (a2D, size);
   for (int i=0; i<size; i++) reverse (a2D[i], size);
   transpose (a2D, size) }
```

4

OOP.MT2.240315.L133

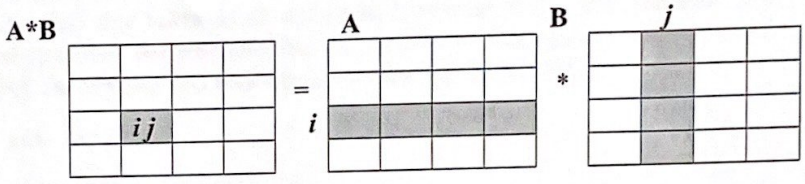
Use the backside, if needed

Name and, if possible, ID#:

Problem 2

Using functions ***transpose()*** from **Problem 1** and ***scalar()*** from below, write a C++ function ***void mult(int *a2D, int *b2D, int *product, int size)*** that takes as its arguments pointers to the first elements of square arrays ***int *a2D*** and ***int *b2D*** of the same specified ***int size***, computes their product and saves it in another square array of the same size, the pointer to the first element of which is given by ***int *product***. Each element p_{ij} in the i^{th} row and j^{th} column of the array ***product** is the scalar product of the i^{th} row of ***a2D** and j^{th} column of ***b2D** and is calculated by the expression: $p_{ij} = \sum_{k=0}^{size-1} a_{ik} b_{kj}$

```
int scalar(int a[], int b[], int length)
{
      int result = 0;
      for (int i = 0; i < length; i++)
            result += a[i] * b[i];
      return result;
}
```




```
void mult
(int *a2D, int *b2D, int *product, int size)
{ for (int i=0;
       i<size;
       i++)

{ for (int j=0
       j<size;
       j++)

{ int tmp=new int [size]; ← returns pointer
  for(int t=0; t<size; t++)
      tmp[t] = a2D[t][j];
  product[i][j] = scalar(a2D[i], tmp, size);
}
}
```

an int

1=2/2

Idx NA

OOP.MT2.2403.05.L33