

Tutorial 3 – Arrays

1. Explain how the addition of 1 to every element of the two dimensional array 'array' is done in the following program. What if the for statement at 'line a' is replaced by this statement:

```
add1(array[0], 3 * 4);
```

```
#include <stdio.h>
void add1(int ar[], int size);
int main()
{
    int array[3][4];
    int h,k;

    for (h = 0; h < 3; h++)
        for (k = 0; k < 4; k++)
            scanf("%d", &array[h][k]);

    for (h = 0; h < 3; h++)                                /* line a */
        add1(array[h], 4);

    for (h = 0; h < 3; h++) {
        for (k = 0; k < 4; k++)
            printf("%10d", array[h][k]);
        putchar('\n');
    }
    return 0;
}
void add1(int ar[], int size)
{
    int k;

    for (k = 0; k < size; k++)
        ar[k]++;
}
```

2. Write a program which will draw the histogram for n integers from 0 to 99. N is input by the user. Each of the n numbers will be generated by calling rand() % 100. The program will consist of two functions (i) to collect the frequency distribution of the numbers (ii) to print the histogram. An example histogram is shown here.

```
0 – 9      | *****
10 – 19    | *****
20 – 29    | *****
30 – 39    | **
.....
90 – 99    | *****
```

3. Write a function that takes a square matrix *ar*, and the array sizes for the rows and columns as parameters, and returns the transpose of the array via call by reference. For example, if the *rowSize* is 4, *colSize* is 4, and the array *ar* is {1,2,3,4, 1,1,2,2, 3,3,4,4, 4,5,6,7}, then the resultant array will be {1,1,3,4, 2,1,3,5, 3,2,4,6, 4,2,4,7}. The function prototype is given below:

```
void transpose2D(int ar[][SIZE], int rowSize, int colSize);
```

4. A square matrix (2-dimensional array of equal dimensions) can be reduced to upper-triangular form by setting each diagonal element to the sum of the original elements in that column and setting to 0s all the elements below the diagonal. For example, the 4-by-4 matrix:

```
4 3 8 6
9 0 6 5
5 1 2 4
9 8 3 7
```

would be reduced to

```
27 3 8 6
0 9 6 5
0 0 5 4
0 0 0 7
```

Write a function `reduceMatrix2D()` to reduce a matrix with dimensions of *rowSize* and *colSize*. The prototype of the function is:

```
void reduceMatrix2D(int ar[][SIZE], int rowSize, int colSize);
```