ASSIGNMENT – ARRAYS

1. (**find2Max1D**) Write a C function `find2Max1D()` that takes an one-dimensional array of integer numbers *`ar`* and *`size`* (>1) as parameters. The function returns the largest and second largest numbers of the array to the calling function through the two pointer parameters *`max1`* and *`max2`* respectively. For example, if the array *`a`*`[]={1,7,8,6,9,4,5,2,3}`, then *`max1`* is 9, and *`max2`* is 8. In this question, you may assume that the input data contains unique integers. The function prototype is given as follows:

```
void find2Max1D(int ar[], int size, int *max1, int *max2);
```

A sample program is given below to test the function:

```
#include <stdio.h>
void find2Max1D(int ar[], int size, int *max1, int *max2);
int main()
{
   int max1,max2;
   int ar[10],size,i;

   printf("Enter array size: \n");
   scanf("%d", &size);
   printf("Enter %d data: \n", size);
   for (i=0; i<size; i++)
      scanf("%d", &ar[i]);
   find2Max1D(ar,size,&max1,&max2);
   printf("Max1: %d\nMax2: %d\n",max1,max2);
   return 0;
}
void find2Max1D(int ar[], int size, int *max1, int *max2)
{
   /* Write your program code here */
}
```

A sample input and output session is given below:

(1) Test Case 1:

```
Enter array size:
5
Enter 5 data:
1 2 3 5 6
Max1: 6
Max2: 5
```

(2) Test Case 2:

```
Enter array size:
6
Enter 6 data:
-4 0 -7 3 2 1
Max1: 3
Max2: 2
```

(3) Test Case 3:

```
Enter array size:
2
Enter 2 data:
6 5
Max1: 6
Max2: 5
```

(4) Test Case 4:

```
Enter array size:
3
Enter 3 data:
-4 -7 -3
Max1: -3
Max2: -4
```

2. **(swapMinMax1D)** Write the C function `swapMinMax1D()` that takes in an array of integers *`ar`* and *`size`* (>1) as parameters, finds the index positions of the largest number and smallest number in the array, swaps the index positions of these two numbers, and passes the array to the calling function via call by reference. For example, if *`ar`* is `{1,2,3,4,5}`, then the resultant array *`ar`* will be `{5,2,3,4,1}` after executing the function. If there are more than one largest or smallest number in the array, we will swap the **last occurrence** of the largest and smallest numbers. For example, if *`ar`* is `{5,2,1,1,8,9,9}`, then the resultant array ar will be `{5,2,1,9,8,9,1}` after executing the function. The function prototype is:

```
void swapMinMax1D(int ar[], int size);
```

A sample program is given below to test the function:

```
#include <stdio.h>
void swapMinMax1D(int ar[], int size);
int main()
{
   int ar[50],i,size;

   printf("Enter array size: \n");
   scanf("%d", &size);
   printf("Enter %d data: \n", size);
   for (i=0; i<size; i++)
      scanf("%d",ar+i);
   swapMinMax1D(ar, size);
   printf("swapMinMax1D(): ");
   for (i=0; i<size; i++)
      printf("%d ",*(ar+i));
   return 0;
}
void swapMinMax1D(int ar[], int size)
{
  /* Write your code here */
}
```

Some sample input and output sessions are given below:

(1) Test Case 1:

```
Enter array size:
5
Enter 5 data:
1 2 3 4 5
swapMinMax1D(): 5 2 3 4 1
```

(2) Test Case 2:

```
Enter array size:
2
Enter 2 data:
5 5
swapMinMax1D(): 5 5
```

(3) Test Case 3:

```
Enter array size:
7
Enter 7 data:
```

```
1 1 1 5 5 5 5
swapMinMax1D(): 1 1 5 5 5 5 1
```

(4) Test Case 4:

```
Enter array size:
9
Enter 9 data:
9 1 1 9 9 5 5 5 5
swapMinMax1D(): 9 1 9 9 1 5 5 5 5
```

3. (**diagonals2D**) Write a C function that accepts a two-dimensional array of integers `ar`, and the array sizes for the rows and columns as parameters, computes the sum of the elements of the two diagonals, and returns the sums to the calling function through the pointer parameters, `sum1` and `sum2`, using call by reference. For example, if the `rowSize` is 3, `colSize` is 3, and the array `ar` is {1,2,3, 1,1,1, 4,3,2}, then `sum1` is computed as 1+1+2=4, and `sum2` is 3+1+4=8. The function prototype is given as follows:

```
void diagonals2D(int ar[][SIZE], int rowSize, int colSize, int
*sum1, int *sum2);
```

A sample program template is given below to test the function:

```
#include <stdio.h>
#define SIZE 10
void diagonals2D(int ar[][SIZE], int rowSize, int colSize, int
*sum1, int *sum2);
int main()
{
   int ar[SIZE][SIZE], rowSize, colSize;
   int i, j, sum1=0, sum2=0;

   printf("Enter row size of the 2D array: \n");
   scanf("%d", &rowSize);
   printf("Enter column size of the 2D array: \n");
   scanf("%d", &colSize);
   printf("Enter the matrix (%dx%d): \n", rowSize, colSize);
   for (i=0; i<rowSize; i++)
      for (j=0; j<colSize; j++)
         scanf("%d", &ar[i][j]);
   diagonals2D(ar, rowSize, colSize, &sum1, &sum2);
   printf("sum1=%d; sum2=%d\n",sum1,sum2);
}
void diagonals2D(int ar[][SIZE], int rowSize, int colSize, int
*sum1, int *sum2)
{
   /* Write your program code here */
}
```

Some sample input and output sessions are given below:

(1) Test Case 1:

```
Enter row size of the 2D array:
3
Enter column size of the 2D array:
3
Enter the matrix (3x3):
1 2 3
1 1 1
4 3 2
sum1=4; sum2=8
```

(2) Test Case 2:

```
Enter row size of the 2D array:
```

```
4
Enter column size of the 2D array:
4
Enter the matrix (4x4):
1 2 3 4
1 1 2 2
2 2 1 1
5 4 3 2
sum1=5; sum2=13
```

(3) Test Case 3:

```
Enter row size of the 2D array:
5
Enter column size of the 2D array:
5
Enter the matrix (4x4):
1 2 3 4 1
1 1 2 2 1
2 2 1 1 1
5 4 3 2 1
5 4 3 2 1
sum1=6; sum2=13
```

4. (**minOfMax2D**) Write a C function `minOfMax2D()` that takes a two-dimensional array matrix of integers *`ar`*, and the array sizes for the rows and columns as parameters. The function returns the minimum of the maximum numbers of each row of the 2-dimensional array `ar`. For example, if the *`rowSize`* is 4, *`colSize`* is 4, and *`ar`* is `{{1,3,5,2}, {2,4,6,8}, {8,6,4,9}, {7,4,3,2}}`, then the maximum numbers will be 5, 8, 9 and 7 for rows 0, 1, 2 and 3 respectively, and the minimum of the maximum numbers will be 5. The prototype of the function is given as follows:

```
int minOfMax2D(int ar[][SIZE], int rowSize, int colSize);
```

A sample program is given below to test the function:

```
#include <stdio.h>
#define SIZE 10
int minOfMax2D(int ar[][SIZE], int rowSize, int colSize);
int main()
{
   int ar[SIZE][SIZE], rowSize, colSize;
   int i,j,min;

   printf("Enter row size of the 2D array: \n");
   scanf("%d", &rowSize);
   printf("Enter column size of the 2D array: \n");
   scanf("%d", &colSize);
   printf("Enter the matrix (%dx%d): \n", rowSize, colSize);
   for (i=0; i<rowSize; i++)
      for (j=0; j<colSize; j++)
         scanf("%d", &ar[i][j]);
   min=minOfMax2D(ar, rowSize, colSize);
   printf("minOfMax2D(): %d\n", min);
   return 0;
}
int minOfMax2D(int ar[][SIZE], int rowSize, int colSize)
{
   /* Write your program code here */
}
```

Some sample input and output sessions are given below:

(1) Test Case 1:

```
Enter row size of the 2D array:
4
Enter column size of the 2D array:
4
Enter the matrix (4x4):
1 2 3 4
2 3 4 5
5 6 7 8
8 10 2 4
minOfMax2D(): 4
```

(2) Test Case 2:

```
Enter row size of the 2D array:
3
Enter column size of the 2D array:
3
Enter the matrix (3x3):
1 -3 3
-3 2 4
3 6 -8
minOfMax2D(): 3
```

(3) Test Case 3:

```
Enter row size of the 2D array:
5
Enter column size of the 2D array:
5
Enter the matrix (4x4):
1 2 3 4 5
2 3 4 5 6
5 6 7 8 9
8 10 2 4 7
2 3 4 5 8
minOfMax2D(): 5
```