Week 4 Lab-Tutorial: Arrays – Suggested Solutions

Lab Questions

Q1: (reverseAr1D)

```c
#include <stdio.h>
void printReverse1(int ar[], int size);
void printReverse2(int ar[], int size);
void reverseAr1D(int ar[], int size);
int main()
{
   int ar[10];
   int size, i;

   printf("Enter array size: \n");
   scanf("%d", &size);
   printf("Enter %d array: \n", size);
   for (i=0; i <= size-1; i++)
      scanf("%d", &ar[i]);
   printReverse1(ar, size);
   printReverse2(ar, size);
   reverseAr1D(ar, size);
   printf("reverseAr1D(): ");
   if (size > 0) {
      for (i=0; i<size; i++)
         printf("%d ", ar[i]);
   }
   return 0;
}
void printReverse1(int ar[], int size)
{
   int i;
   printf("printReverse1(): ");
   if (size > 0) {
      for (i=size-1; i>=0; i--)
         printf("%d ", ar[i]);
   }
   printf("\n");
}
void printReverse2(int ar[], int size)
{
   int i;
   printf("printReverse2(): ");
   if (size > 0) {
      for (i=size-1; i>=0; i--)
         printf("%d ", *(ar+i));
   }
   printf("\n");
}
/* reverseAr reverses the array contents and passes that back to the
calling function */
void reverseAr1D(int ar[], int size)
{
```

```
   int i, temp;
   if (size > 0) {
      for (i=0; i<size/2; i++){
         temp = ar[i];
         ar[i] = ar[size-i-1];
         ar[size-i-1] = temp;
      }
   }
}
```

Q2: (findAr1D)

```
#include <stdio.h>
int findAr1D(int size, int ar[], int target);
int main()
{
   int ar[20];
   int size, i, target, result = 888;

   printf("Enter array size: \n");
   scanf("%d", &size);
   printf("Enter %d data: \n", size);
   for (i=0; i<=size-1; i++)
      scanf("%d", &ar[i]);
   printf("Enter the target number: \n");
   scanf("%d", &target);
   result = findAr1D(size, ar, target);
   if (result == -1)
      printf("findAr1D(): Not found\n");
   else
      printf("findAr1D(): %d", result);
   return 0;
}
int findAr1D(int size, int ar[], int target)
{
   int j;
   for (j = 0; j < size; j++)
      if (ar[j] == target)
         return j;
   return -1;
}
/* another version */
/*
int findAr1D(int size, int ar[], int target)
{
   int j;
   for (j = 0; j < size; j++)
      if (*(ar+j) == target)
         return j;
   return -1;
}
*/
```

Q3: (swap2RowsCols2D)

```c
#include <stdio.h>
#define SIZE 3
void swap2Rows(int ar[][SIZE], int r1, int r2);
void swap2Cols(int ar[][SIZE], int c1, int c2);
void display(int ar[][SIZE]);
int main()
{
   int array[SIZE][SIZE];
   int row1, row2, col1, col2;
   int i,j;
   int choice;

   printf("Select one of the following options: \n");
   printf("1: getInput()\n");
   printf("2: swap2Rows()\n");
   printf("3: swap2Cols()\n");
   printf("4: display()\n");
   printf("5: exit()\n");
   do {
      printf("Enter your choice: \n");
      scanf("%d", &choice);
      switch (choice) {
         case 1:
            printf("Enter the matrix (3x3): \n");
            for (i=0; i<SIZE; i++)
               for (j=0; j<SIZE; j++)
                  scanf("%d", &array[i][j]);
            break;
         case 2:
            printf("Enter two rows for swapping: \n");
            scanf("%d %d", &row1, &row2);
            swap2Rows(array, row1, row2);
            printf("The new array is: \n");
            display(array);
            break;
         case 3:
            printf("Enter two columns for swapping: \n");
            scanf("%d %d", &col1, &col2);
            swap2Cols(array, col1, col2);
            printf("The new array is: \n");
            display(array);
            break;
         case 4:
            display(array);
            break;
      }
   } while (choice < 5);
   return 0;
}
void display(int ar[][SIZE])
{
   int l,m;
   for (l = 0; l < SIZE; l++) {
      for (m = 0; m < SIZE; m++)
         printf("%d ", ar[l][m]);
      printf("\n");
   }
}
void swap2Rows(int ar[][SIZE], int r1, int r2)
```

```
/* swaps row ar[r1] with row ar[r2] */
{
   int temp;
   int n;

   for(n = 0; n < SIZE; n++) {
      temp = ar[r1][n] ;
      ar[r1][n] = ar[r2][n];
      ar[r2][n] = temp;
   }
}
void swap2Cols(int ar[][SIZE], int c1, int c2)
/* swaps column ar[][c1] with column ar[][c2] */
{
   int temp;
   int n;

   for(n = 0; n < SIZE; n++) {
      temp = ar[n][c1] ;
      ar[n][c1] = ar[n][c2];
      ar[n][c2] = temp;
   }
}
```

Q4: (reduceMatrix2D)

```
#include <stdio.h>
#define SIZE 10
void reduceMatrix2D(int ar[][SIZE], int rowSize, int colSize);
void display(int ar[][SIZE], int rowSize, int colSize);
int main()
{
   int ar[SIZE][SIZE], rowSize, colSize;
   int i,j;
   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]);

   reduceMatrix2D(ar, rowSize, colSize);
   printf("reduceMatrix2D(): \n");
   display(ar, rowSize, colSize);
   return 0;
}
void display(int ar[][SIZE], int rowSize, int colSize)
{
   int l,m;
   for (l = 0; l < rowSize; l++) {
      for (m = 0; m < colSize; m++)
         printf("%d ", ar[l][m]);
      printf("\n");
   }
}
void reduceMatrix2D(int ar[][SIZE], int rowSize, int colSize)
{
   int i, j, sum; // i for row, j for column
   /* for each column */
   for (j = 0; j < colSize; j++){
```

```
      sum = 0;
      // process the row below matrix[j][j] of the column
      for (i = j+1; i < rowSize; i++){
         sum += ar[i][j];
         ar[i][j] = 0;
      }
      ar[j][j] += sum;
   }
}
```