3 Versions of 2D Array Sorting with Bubble Sort (Simplified Version)

GROUP 3: BSIT 1-4
Members:
Agad, Jiro Laurenz
Alpornon, Christine Julia
Bacolod, Mikaila Jhay
Buga-ay, Carl Tristan
Cheng, Xian Hui
Del Rosario, Kyle Ferell
Edusma, Marie Cris
Esteban, Anthony James
Miguel, Rafael Louie
Modelo, John Vincent
Mortel, Meg Edelviese
Orde, Cyrus Jezter
Pascual, Ian Nevri
Robles, Aira Mae
Rosarda, Jeroises Israel
Sebastian, Brian May

Pointer version (*p)

```
#include <stdio.h>
#include <malloc.h>

#define ROWS 3
#define COLS 3

void BubbleSortRows(double *p, int ifAscend);
void BubbleSortWhole (double *p, int ifAscend);
void GetArray(double *p);
void printBoard(double *p);

int i, j, k;
int main(){
        double **numbers = (double**)malloc(ROWS * sizeof(double*));
         int ifAscend;
         for (i = 0; i < ROWS; i++)
                *(numbers + i) = (double*)malloc(COLS * sizeof(double));
        double *p = numbers[0];

        GetArray(p);

        printf(“Enter [1] for Ascending and [2] for Descending”);
        scanf(“%d”, &ifAscend);

        printf("\n1. Per Row Sorting: \n");
        BubbleSortRows(p, ifAscend);
        printBoard(p);

        printf("\n2. Whole Board Sorting: \n");
        BubbleSortWhole(p, ifAscend);
        printBoard(p);

return 0;
}
```

```
void BubbleSortRows(double *p, int ifAscend) {
    for (i = 0; i < ROWS; i++) {
        for (j = 0; j < COLS - 1; j++) {
            for (k = 0; k < COLS - j - 1; k++) {

                double *current = (p + i * COLS + k);
                double *next = (p + i * COLS + k + 1);

                if (*(p + i * COLS + k) > *(p + i * COLS + k + 1) && ifAscend == 1 || *(p + i * COLS + k) < *(p + i * COLS + k + 1) && ifAscend == 2){
                    double temp = *(p + i * COLS + k);
                    *(p + i * COLS + k) = *(p + i * COLS + k + 1);
                    *(p + i * COLS + k + 1) = temp;
                }
            }
        }
    }
}

void BubbleSortWhole(double *p, int ifAscend) {
    int totalElements = ROWS * COLS;

    for (i = 0; i < totalElements; i++) {
        for (j = 0; j < totalElements - i - 1; j++) {
            double* current = p + j;
            double* next = p + j + 1;

            if ((*current > *next && ifAscend == 1) || (*current < *next && ifAscend == 2)) {
                double temp = *current;
                *current = *next;
                *next = temp;
            }
        }
    }
}


void GetArray(double *p) {
    for (i = 0; i < ROWS; i++) {
        printf("%s%sROW: %d \n%s", BOLD, YELLOW, i + 1, RESET);
        for (j = 0; j < COLS; j++) {
            double input;
            printf("Enter any number for [COLUMN %d]: ", j + 1);
            input = getValidNumber(input, j);
            *(p + i * COLS + j) = input;
        }
        printf("\n");
    }
}

void printBoard(double *p) {
    for (i = 0; i < ROWS; i++) {
        if (i == 0 || i == 2)
            printf("\n+----------------------+\n");
        else
            printf("\n------------------------\n");
        for (j = 0; j < COLS; j++) {
            printf("%s%*s%*g%*s%s", j < COLS ? "|" : "",
                *(p + i * COLS + j) < 10 ? 3 : (*(p + i * COLS + j) < 100) ? 3 : 2, "",
                *(p + i * COLS + j) < 10 ? 1 : (*(p + i * COLS + j) < 100) ? 2 : 3,
                *(p + i * COLS + j),
                *(p + i * COLS + j) < 10 ? 3 : (*(p + i * COLS + j) < 100) ? 2 : 2, "",
                j == COLS - 1 ? "|" : "");
        }
    }
    printf("\n+----------------------+\n");
}
```

OUTPUT Pointer Version

Ascending

```
==========================================
|| 2d Array Sorter: Bubble Sort Edition ||
==========================================

ROW: 1
Enter any number for [COLUMN 1]: 9
Enter any number for [COLUMN 2]: 8
Enter any number for [COLUMN 3]: 7

ROW: 2
Enter any number for [COLUMN 1]: 6
Enter any number for [COLUMN 2]: 5
Enter any number for [COLUMN 3]: 4

ROW: 3
Enter any number for [COLUMN 1]: 3
Enter any number for [COLUMN 2]: 2
Enter any number for [COLUMN 3]: 1

============================================================

ENTER [1] FOR ASCENDING OR [2] FOR DESCENDING: 1
You entered: Ascending

Please enter any key to continue.

<== BEFORE SORTING ==>

+-----------------------+
|   9   |   8   |   7   |
-------------------------
|   6   |   5   |   4   |
+-----------------------+
|   3   |   2   |   1   |
+-----------------------+

============================================================

Sorting Complete

<== BEFORE SORTING ==>

+-----------------------+
|   9   |   8   |   7   |
-------------------------
|   6   |   5   |   4   |
+-----------------------+
|   3   |   2   |   1   |
+-----------------------+

============================================================

Sorting Complete

<== AFTER SORTING ==>

1. Per Row Sorting:

+-----------------------+
|   7   |   8   |   9   |
-------------------------
|   4   |   5   |   6   |
+-----------------------+
|   1   |   2   |   3   |
+-----------------------+

2. Whole Board Sorting:

+-----------------------+
|   1   |   2   |   3   |
-------------------------
|   4   |   5   |   6   |
+-----------------------+
|   7   |   8   |   9   |
+-----------------------+

============================================================

Try Again?
Enter 'Y' for Yes or 'N' for No:
```

Descending

```
==========================================
|| 2d Array Sorter: Bubble Sort Edition ||
==========================================

ROW: 1
Enter any number for [COLUMN 1]: 1
Enter any number for [COLUMN 2]: 2
Enter any number for [COLUMN 3]: 3

ROW: 2
Enter any number for [COLUMN 1]: 4
Enter any number for [COLUMN 2]: 5
Enter any number for [COLUMN 3]: 6

ROW: 3
Enter any number for [COLUMN 1]: 7
Enter any number for [COLUMN 2]: 8
Enter any number for [COLUMN 3]: 9

============================================================

ENTER [1] FOR ASCENDING OR [2] FOR DESCENDING: 2
You entered: Descending

Please enter any key to continue.

<== BEFORE SORTING ==>

+-----------------------+
|   1   |   2   |   3   |
-------------------------
|   4   |   5   |   6   |
+-----------------------+
|   7   |   8   |   9   |
+-----------------------+

============================================================

Sorting Complete
```

```
<== BEFORE SORTING ==>

+-----------------------+
|   1   |   2   |   3   |
-------------------------
|   4   |   5   |   6   |
+-----------------------+
|   7   |   8   |   9   |
+-----------------------+

============================================================

Sorting Complete

<== AFTER SORTING ==>

1. Per Row Sorting:

+-----------------------+
|   3   |   2   |   1   |
-------------------------
|   6   |   5   |   4   |
+-----------------------+
|   9   |   8   |   7   |
+-----------------------+

2. Whole Board Sorting:

+-----------------------+
|   9   |   8   |   7   |
-------------------------
|   6   |   5   |   4   |
+-----------------------+
|   3   |   2   |   1   |
+-----------------------+

============================================================

Try Again?
Enter 'Y' for Yes or 'N' for No:
```

Version 1(double **numbers)

```
#include <stdio.h>
#include <malloc.h>

#define ROWS 3
#define COLS 3

void BubbleSortRows(double **numbers, int ifAscend);
void BubbleSortWhole (double **numbers, int ifAscend);
void GetArray(double **numbers);
void printBoard(double **numbers);

int i, j, k;
int main(){
        double **numbers = (double**)malloc(ROWS * sizeof(double*));
         int ifAscend;
         for (i = 0; i < ROWS; i++)
                *(numbers + i) = (double*)malloc(COLS * sizeof(double));

        GetArray(numbers);

        printf(“Enter [1] for Ascending and [2] for Descending”);
        scanf(“%d”, &ifAscend);

        printf("\n1. Per Row Sorting: \n");
        BubbleSortRows(numbers, ifAscend);
        printBoard(numbers);

        printf("\n2. Whole Board Sorting: \n");
        BubbleSortWhole(numbers, ifAscend);
        printBoard(numbers);

return 0;
}

void BubbleSortRows (double **numbers, int ifAscend){
   for (i = 0; i < ROWS; i++) {
      for (j = 0; j < COLS - 1; j++) {
         for (k = 0; k < COLS - j - 1; k++) {
            // Check the sorting order and compare adjacent elements
            if ((*(numbers[i]) + k > *(numbers[i] + k + 1) && ifAscend == 1) || (*(numbers[i] + k) < *(numbers[i] + k + 1) && ifAscend == 2)) {
               // Swap the elements if they are in the wrong order
               double temp = *(numbers[i] + k);
               *(numbers[i] + k) = *(numbers[i] + k + 1);
               *(numbers[i] + k + 1) = temp;
            }
         }
      }
```

```
    }
}

void BubbleSortWhole(double **numbers, int ifAscend) {
    int totalElements = ROWS * COLS;

    for (i = 0; i < totalElements; i++) {
        for (j = 0; j < totalElements - i - 1; j++) {
            double *current = numbers[j / COLS] + j % COLS;
            double *next = numbers [(j + 1) / COLS] + (j + 1) % COLS;

            if ((*current > *next && ifAscend == 1) || (*current < *next && ifAscend == 2)) {
                // Swap the elements if they are in the wrong order
                double temp = *current;
                *current = *next;
                *next = temp;
            }
        }
    }
}

void GetArray(double **numbers){
    for (i = 0; i < ROWS; i++){
        printf("%s%sROW: %d \n%s", BOLD, YELLOW, i + 1, RESET);
        for (j = 0; j < COLS; j++){
            double input;
            printf("Enter any number for [COLUMN %d]: ", j + 1);
            input = getValidNumber(input, j);
            *(numbers[i] + j) = input;
        }
        printf("\n");
    }
}

void printBoard(double **numbers){
    for(i = 0; i <  ROWS; i++){
        if(i == 0 || i == 2)
            printf("\n+----------------------+\n");
        else
            printf("\n------------------------\n");
        for (j = 0; j < COLS; j++){
            printf("%s%*s%*g%*s%s", j < COLS ? "|" : "",
                            (*(numbers[i] + j)) < 10 ? 3 : ((*(numbers[i]+ j)) < 100) ? 3 : 2, "",
                            (*(numbers[i] + j)) < 10 ? 1 : ((*(numbers[i]+ j)) < 100) ? 2 : 3,
                            *(numbers[i] + j),
                            (*(numbers[i] + j)) < 10 ? 3 : ((*(numbers[i]+ j)) < 100) ? 2 : 2, "",
                            j == COLS - 1 ? "|" : "");

        }
    }
    printf("\n+----------------------+\n");
}
```

OUTPUT Version 1

Ascending

```
========================================
|| 2d Array Sorter: Bubble Sort Edition ||
========================================

ROW: 1
Enter any number for [COLUMN 1]: 9
Enter any number for [COLUMN 2]: 8
Enter any number for [COLUMN 3]: 7

ROW: 2
Enter any number for [COLUMN 1]: 6
Enter any number for [COLUMN 2]: 5
Enter any number for [COLUMN 3]: 4

ROW: 3
Enter any number for [COLUMN 1]: 3
Enter any number for [COLUMN 2]: 2
Enter any number for [COLUMN 3]: 1

============================================================

ENTER [1] FOR ASCENDING OR [2] FOR DESCENDING: 1
You entered: Ascending

Please enter any key to continue.

<== BEFORE SORTING ==>

+-----------------------+
|   9   |   8   |   7   |
-------------------------
|   6   |   5   |   4   |
+-----------------------+
|   3   |   2   |   1   |
+-----------------------+

============================================================

Sorting Complete

<== BEFORE SORTING ==>

+-----------------------+
|   9   |   8   |   7   |
-------------------------
|   6   |   5   |   4   |
+-----------------------+
|   3   |   2   |   1   |
+-----------------------+

============================================================

Sorting Complete

<== AFTER SORTING ==>

1. Per Row Sorting:

+-----------------------+
|   7   |   8   |   9   |
-------------------------
|   4   |   5   |   6   |
+-----------------------+
|   1   |   2   |   3   |
+-----------------------+

2. Whole Board Sorting:

+-----------------------+
|   1   |   2   |   3   |
-------------------------
|   4   |   5   |   6   |
+-----------------------+
|   7   |   8   |   9   |
+-----------------------+

============================================================

Try Again?
Enter 'Y' for Yes or 'N' for No:
```

Descending

```
========================================
|| 2d Array Sorter: Bubble Sort Edition ||
========================================

ROW: 1
Enter any number for [COLUMN 1]: 1
Enter any number for [COLUMN 2]: 2
Enter any number for [COLUMN 3]: 3

ROW: 2
Enter any number for [COLUMN 1]: 4
Enter any number for [COLUMN 2]: 5
Enter any number for [COLUMN 3]: 6

ROW: 3
Enter any number for [COLUMN 1]: 7
Enter any number for [COLUMN 2]: 8
Enter any number for [COLUMN 3]: 9

============================================================

ENTER [1] FOR ASCENDING OR [2] FOR DESCENDING: 2
You entered: Descending

Please enter any key to continue.

<== BEFORE SORTING ==>

+-----------------------+
|   1   |   2   |   3   |
-------------------------
|   4   |   5   |   6   |
+-----------------------+
|   7   |   8   |   9   |
+-----------------------+

============================================================

Sorting Complete
```

```
<== BEFORE SORTING ==>

+-----------------------+
|   1   |   2   |   3   |
-------------------------
|   4   |   5   |   6   |
+-----------------------+
|   7   |   8   |   9   |
+-----------------------+

============================================================

Sorting Complete

<== AFTER SORTING ==>

1. Per Row Sorting:

+-----------------------+
|   3   |   2   |   1   |
-------------------------
|   6   |   5   |   4   |
+-----------------------+
|   9   |   8   |   7   |
+-----------------------+

2. Whole Board Sorting:

+-----------------------+
|   9   |   8   |   7   |
-------------------------
|   6   |   5   |   4   |
+-----------------------+
|   3   |   2   |   1   |
+-----------------------+

============================================================

Try Again?
Enter 'Y' for Yes or 'N' for No:
```

Version 2(double **numbers)

```
#include <stdio.h>
#include <malloc.h>

#define ROWS 3
#define COLS 3

void BubbleSortRows(double **numbers, int ifAscend);
void BubbleSortWhole (double **numbers, int ifAscend);
void GetArray(double **numbers);
void printBoard(double **numbers);

int i, j, k;
int main(){
        double **numbers = (double**)malloc(ROWS * sizeof(double*));
         int ifAscend;
         for (i = 0; i < ROWS; i++)
                *(numbers + i) = (double*)malloc(COLS * sizeof(double));

        GetArray(numbers);

        printf(“Enter [1] for Ascending and [2] for Descending”);
        scanf(“%d”, &ifAscend);

        printf("\n1. Per Row Sorting: \n");
        BubbleSortRows(numbers, ifAscend);
        printBoard(numbers);

        printf("\n2. Whole Board Sorting: \n");
        BubbleSortWhole(numbers, ifAscend);
        printBoard(numbers);

return 0;
}

void BubbleSortRows (double **numbers, int ifAscend){
   for (i = 0; i < ROWS; i++) {
      for (j = 0; j < COLS - 1; j++) {
         for (k = 0; k < COLS - j - 1; k++) {
            // Check the sorting order and compare adjacent elements
            double *current = *(numbers + i) + k;
            double *next = *(numbers + i) + k + 1;

            if ((*(*(numbers + i) + k) > *(*(numbers + i) + k + 1) && ifAscend == 1) || (*(*(numbers + i) + k) < *(*(numbers + i) + k + 1) && ifAscend
== 2)) {
               // Swap the elements if they are in the wrong order
               double temp = *current;
               *current = *next;
               *next = temp;
            }
         }
      }
   }
}

void BubbleSortWhole(double **numbers, int ifAscend) {
   int totalElements = ROWS * COLS;

   for (i = 0; i < totalElements; i++) {
      for (j = 0; j < totalElements - i - 1; j++) {
         double *current = *(numbers + (j / COLS)) + j % COLS;
         double *next = *(numbers + ((j + 1) / COLS)) + (j + 1) % COLS;

         if ((*current > *next && ifAscend == 1) || (*current < *next && ifAscend == 2)) {
            double temp = *current;
            *current = *next;
            *next = temp;
         }
      }
   }
}
```

```
void GetArray(double **numbers){
    for (i = 0; i < ROWS; i++){
        printf("%s%sROW: %d \n%s", BOLD, YELLOW, i + 1, RESET);
        for (j = 0; j < COLS; j++){
            double input;
            printf("Enter any number for [COLUMN %d]: ", j + 1);
            input = getValidNumber(input, j);
            *(*(numbers + i) + j) = input;
        }
        printf("\n");
    }
}

void printBoard(double **numbers){
    for(i = 0; i <  ROWS; i++){
        if(i == 0 || i == 2)
            printf("\n+----------------------+\n");
        else
            printf("\n------------------------\n");
        for (j = 0; j < COLS; j++){
            printf("%s%*s%*g%*s%s", j < COLS ? "|" : "",
                        (*(*(numbers + i) + j)) < 10 ? 3 : ((*(*(numbers + i) + j)) < 100) ? 3 : 2, "",
                        *(*(numbers + i) + j),
                        (*(*(numbers + i) + j)) < 10 ? 3 : ((*(*(numbers + i) + j)) < 100) ? 2 : 2, "",
                        j == COLS - 1 ? "|" : "");

        }
    }
    printf("\n+----------------------+\n");
}
```

OUTPUT Version 2

Ascending

```
========================================
|| 2d Array Sorter: Bubble Sort Edition ||
========================================

ROW: 1
Enter any number for [COLUMN 1]: 9
Enter any number for [COLUMN 2]: 8
Enter any number for [COLUMN 3]: 7

ROW: 2
Enter any number for [COLUMN 1]: 6
Enter any number for [COLUMN 2]: 5
Enter any number for [COLUMN 3]: 4

ROW: 3
Enter any number for [COLUMN 1]: 3
Enter any number for [COLUMN 2]: 2
Enter any number for [COLUMN 3]: 1

============================================================

ENTER [1] FOR ASCENDING OR [2] FOR DESCENDING: 1
You entered: Ascending

Please enter any key to continue.

<== BEFORE SORTING ==>

+----------------------+
|   9   |   8   |   7   |
------------------------
|   6   |   5   |   4   |
+----------------------+
|   3   |   2   |   1   |
+----------------------+

============================================================

Sorting Complete
```

```
<== BEFORE SORTING ==>

+-----------------------+
|   9   |   8   |   7   |
-------------------------
|   6   |   5   |   4   |
+-----------------------+
|   3   |   2   |   1   |
+-----------------------+

============================================================

Sorting Complete

<== AFTER SORTING ==>

1. Per Row Sorting:

+-----------------------+
|   7   |   8   |   9   |
-------------------------
|   4   |   5   |   6   |
+-----------------------+
|   1   |   2   |   3   |
+-----------------------+

2. Whole Board Sorting:

+-----------------------+
|   1   |   2   |   3   |
-------------------------
|   4   |   5   |   6   |
+-----------------------+
|   7   |   8   |   9   |
+-----------------------+

============================================================

Try Again?
Enter 'Y' for Yes or 'N' for No: |
```

Descending

```
==========================================
|| 2d Array Sorter: Bubble Sort Edition ||
==========================================

ROW: 1
Enter any number for [COLUMN 1]: 1
Enter any number for [COLUMN 2]: 2
Enter any number for [COLUMN 3]: 3

ROW: 2
Enter any number for [COLUMN 1]: 4
Enter any number for [COLUMN 2]: 5
Enter any number for [COLUMN 3]: 6

ROW: 3
Enter any number for [COLUMN 1]: 7
Enter any number for [COLUMN 2]: 8
Enter any number for [COLUMN 3]: 9

============================================================

ENTER [1] FOR ASCENDING OR [2] FOR DESCENDING: 2
You entered: Descending

Please enter any key to continue.

<== BEFORE SORTING ==>

+-----------------------+
|   1   |   2   |   3   |
-------------------------
|   4   |   5   |   6   |
+-----------------------+
|   7   |   8   |   9   |
+-----------------------+

============================================================

Sorting Complete
```

```
<== BEFORE SORTING ==>

+-----------------------+
|   1   |   2   |   3   |
-------------------------
|   4   |   5   |   6   |
+-----------------------+
|   7   |   8   |   9   |
+-----------------------+

============================================================

Sorting Complete

<== AFTER SORTING ==>

1. Per Row Sorting:

+-----------------------+
|   3   |   2   |   1   |
-------------------------
|   6   |   5   |   4   |
+-----------------------+
|   9   |   8   |   7   |
+-----------------------+

2. Whole Board Sorting:

+-----------------------+
|   9   |   8   |   7   |
-------------------------
|   6   |   5   |   4   |
+-----------------------+
|   3   |   2   |   1   |
+-----------------------+

============================================================

Try Again?
Enter 'Y' for Yes or 'N' for No:
```