University of Mila

Faculty of Science and Technology

Department of Process Engineering

Practical Work 04

Solutions

Course: Introduction to Programming

Level: 1st Year ST - ENG & LMD

Semester 02

By:

Dr. Farouk KECITA

Academic Year: 2025/2026

Solution 01: Reading and Printing Array Elements

Question: Write a C program that reads n elements from two 1D arrays, arr_1[n] and arr_2[n], entered by the user. Then, print the elements of both arrays separately.

Program:

```c
#include <stdio.h>

int main() {
    int n1, n2, i;
    int arr1[100], arr2[100];  // Declare arrays with maximum size
       100

    // Read first array
    printf("Enter size of the arr1: ");
    scanf("%d", &n1);
    printf("Enter elements in arr1: \n");
    for(i = 0; i < n1; i++) {
        scanf("%d", &arr1[i]);  // Store elements in first array
    }

    // Read second array
    printf("Enter size of the arr2: ");
    scanf("%d", &n2);
    printf("Enter elements in arr2: \n");
    for(i = 0; i < n2; i++) {
        scanf("%d", &arr2[i]);  // Store elements in second array
    }

    // Display first array
    printf("\n1st array :\n");
    for(i = 0; i < n1; i++) {
        printf("arr1[%d] = %d\n", i, arr1[i]);
    }

    // Display second array
    printf("\n2nd array :\n");
    for(i = 0; i < n2; i++) {
        printf("arr2[%d] = %d\n", i, arr2[i]);
    }

    return 0;  // Indicate successful program execution
}
```

Notes:

- Arrays are declared with a maximum size of 100 to avoid dynamic allocation issues.
- The `for` loop is used to iterate through array elements for both input and output.
- The program assumes the user will enter valid array sizes.

Sample Output

```
Enter size of the arr1: 5
Enter elements in arr1:
21
10
15
60
32

Enter size of the arr2: 4
Enter elements in arr2:
11
10
9
31

1st array :
arr1[0] = 21
arr1[1] = 10
arr1[2] = 15
arr1[3] = 60
arr1[4] = 32

2nd array :
arr2[0] = 11
arr2[1] = 10
arr2[2] = 9
arr2[3] = 31
```

Solution 02: Merging Two Arrays

Question: Write a C program to read elements into two 1D arrays and merge them into a third array.

Program:

```c
#include <stdio.h>

int main() {
    int i, n1, n2;
    int arr1[100], arr2[100], arr3[200];//arr3size= arr1size+arr2size

    // Read first array
    printf("Enter size of the 1st array: ");
    scanf("%d", &n1);
    printf("Enter elements in array:\n");
    for(i = 0; i < n1; i++) {
        scanf("%d", &arr1[i]);
    }

```

```c
    // Read second array
    printf("Enter size of the 2nd array: ");
    scanf("%d", &n2);
    printf("Enter elements in array:\n");
    for(i = 0; i < n2; i++) {
        scanf("%d", &arr2[i]);
    }

    // Merging arrays: copy all elements from arr1 and arr2 to arr3
    for(i = 0; i < n1 + n2; i++) {
        if(i < n1)
            arr3[i] = arr1[i];        // Copy from first array
        else
            arr3[i] = arr2[i - n1];//Copy from second array (adjust
                                                                index)
    }

    // Display first array
    printf("\n1st array:\n");
    for(i = 0; i < n1; i++) {
        printf("arr1[%d] = %d\t", i, arr1[i]);
    }

    // Display second array
    printf("\n\n2nd array:\n");
    for(i = 0; i < n2; i++) {
        printf("arr2[%d] = %d\t", i, arr2[i]);
    }

    // Display merged array
    printf("\n\n3rd array (merged):\n");
    for(i = 0; i < n1 + n2; i++) {
        printf("arr3[%d] = %d\n", i, arr3[i]);
    }

    return 0;
}
```

Notes:

- The third array `arr3` has a size equal to the sum of the sizes of the first two arrays.
- In the merging process, we first copy all elements from `arr1`, then copy elements from `arr2`.
- When copying from `arr2`, we use the index `i - n1` to access the correct element.

Sample Output

```
Enter size of the 1st array: 4
Enter elements in array:
5
7
8
9

Enter size of the 2nd array: 3
Enter elements in array:
10
20
50

1st array:
arr1[0] = 5 arr1[1] = 7 arr1[2] = 8 arr1[3] = 9

2nd array:
arr2[0] = 10 arr2[1] = 20 arr2[2] = 50

3rd array (merged):
arr3[0] = 5
arr3[1] = 7
arr3[2] = 8
arr3[3] = 9
arr3[4] = 10
arr3[5] = 20
arr3[6] = 50
```

Solution 03: Split Array into Even and Odd

Question: Write a C program to split a 1D array into two separate arrays based on even and odd values.

Program:

```c
#include <stdio.h>
int main() {
    long int arr[100], odarr[100], evarr[100];
    int i, j = 0, k = 0, s;
    printf("Enter the size of the array\n");
    scanf("%d", &s);
    printf("Enter the elements of the array\n");
    for(i = 0; i < s; i++) {
        scanf("%ld", &arr[i]);
    }
    // Split array: separate even and odd elements
    for(i = 0; i < s; i++) {
        if(arr[i] % 2 == 0) {          // Check if element is even
            evarr[j] = arr[i];        // Store in even array
```

```
            j++;                        // Increment even array index
        }
        else {                          // Element is odd
            odarr[k] = arr[i];          // Store in odd array
            k++;                       // Increment odd array index
        }
    }
    // Display odd array
    printf("\nThe elements of the Odd array are:\n");
    for(i = 0; i < k; i++) {
        printf("odarr[%d] = %ld\n", i, odarr[i]);
    }
    // Display even array
    printf("\nThe elements of the Even array are:\n");
    for(i = 0; i < j; i++) {
        printf("evarr[%d] = %ld\n", i, evarr[i]);
    }
    return 0;
}
```

Notes:

- We use two separate counters: `j` for the even array and `k` for the odd array.
- Even condition: `arr[i] % 2 == 0` (remainder when divided by 2 equals 0).
- Odd numbers are those that do not satisfy the even condition.

Sample Output

```
Enter the size of the array
6
Enter the elements of the array
10
11
12
13
14
15

The elements of the Odd array are:
odarr[0] = 11
odarr[1] = 13
odarr[2] = 15

The elements of the Even array are:
evarr[0] = 10
evarr[1] = 12
evarr[2] = 14
```

Solution 04 (Homework): Frequency of Elements

Question: Write a C program to count the frequency of each element in a 1D array.

Program :

```c
#include <stdio.h>
int main() {
    int arr[100], freq[100], i, j, size, c = 0;
    printf("Enter size of the array: ");
    scanf("%d", &size);

    printf("Enter elements in array:\n");
    for(i = 0; i < size; i++) {
        scanf("%d", &arr[i]);
    }

    // Calculate frequency of each element
    for(i = 0; i < size; i++) {
        c = 1;  // Initialize counter to 1 (current element)

        if(arr[i] != -1) {  // If element not already counted
            // Compare with remaining elements
            for(j = i + 1; j < size; j++) {
                if(arr[i] == arr[j]) {
                    c++;             // Increment counter
                    arr[j] = -1;   // Mark as counted (special value)
                }
            }
            freq[i] = c;  // Store frequency
        }
    }

    // Display frequencies
    printf("\nFrequency of elements:\n");
    for(i = 0; i < size; i++) {
        if(arr[i] != -1) {// Only display elements that weren't
           marked
            printf("Frequency of %d is %d\n", arr[i], freq[i]);
        }
    }

    return 0;
}
```

Notes:

- We use the value `-1` to mark elements that have already been counted.
- For each element, we search the rest of the array for duplicates.
- The `freq` array stores the frequency of each element.
- This method modifies the original array by setting duplicate elements to `-1`.

Sample Output

```
Enter size of the array: 10
Enter elements in array:
12
15
12
17
16
12
15
20
18
20

Frequency of elements:
Frequency of 12 is 3
Frequency of 15 is 2
Frequency of 17 is 1
Frequency of 16 is 1
Frequency of 20 is 2
Frequency of 18 is 1
```

Solution 05: Array Sum and Average

Question: Write a C program to store integers entered by the user in a one-dimensional array, then compute the sum and average of the elements.

Program :

```c
#include <stdio.h>

int main() {
    int arr[100], s, i, sum = 0;
    float avg;

    // Get the array size input from user
    printf("Enter array size:\n");
    scanf("%d", &s);

    // Get all elements of array
    printf("Enter array elements:\n");
    for(i = 0; i < s; i++) {
        scanf("%d", &arr[i]);
    }

    // Calculate sum of all elements
    for(i = 0; i < s; i++) {
        sum = sum + arr[i];  // Add each element to sum
    }

```

```c
    // Calculate average (typecast sum to float for accurate
       division)
    avg = (float)sum / s;

    // Print the result
    printf("Sum of all elements in arr = %d\n", sum);
    printf("Average of array values is: %.2f\n", avg);  //

    return 0;
}
```

Notes:

- The variable `sum` stores the total of all elements and is initialized to 0.
- Notice the use of `(float)sum` to convert the sum to a floating-point number before division.
- This ensures accurate average calculation (e.g., 15.20 instead of 15).
- The format specifier `%.2f` prints only 2 decimal places.

Sample Output

```
Enter array size:
10
Enter array elements:
13
12
15
20
15
16
16
17
18
10

Sum of all elements in arr = 152
Average of array values is: 15.20
```

Solution 06: Dot Product of Two Vectors

Question: Write a C program to compute the dot product of two 1D arrays (vectors).
Program :

```c
#include <stdio.h>

int main() {
    int i, size1, size2, dotp = 0;

    printf("Enter the size of vectors vect_A and vect_B:\n");
```

```c
    scanf("%d%d", &size1, &size2);

    // Check if vectors have the same size
    if(size1 != size2) {
        printf("Warning!! The size of the two vectors is not the
            same.\n");
        return 0;  // Exit program
    }

    int vect_A[size1],vect_B[size2];// Declare vectors with given
                                                              sizes

    printf("Enter the elements of vect_A:\n");
    for(i = 0; i < size1; i++) {
        scanf("%d", &vect_A[i]);
    }

    printf("Enter the elements of vect_B:\n");
    for(i = 0; i < size2; i++) {
        scanf("%d", &vect_B[i]);
    }

    // Calculate dot product: sum of (a[i] * b[i])
    for(i = 0; i < size2; i++) {
        dotp += vect_A[i] * vect_B[i];  // Add product to dotp
    }

    printf("The inner product of these two vectors is %d\n", dotp);

    return 0;
}
```

Notes:

- The dot product is only defined for vectors of the same length.
- Therefore, we first check if `size1` equals `size2`.
- If the sizes are different, a warning is displayed and the program exits.
- Mathematical formula: $\mathbf{A} \cdot \mathbf{B} = \sum_{i=1}^{n} A_i \times B_i$

Output Run 1 - Equal Sizes

```
Enter the size of vectors vect_A and vect_B:
4 4
Enter the elements of vect_A:
1
2
3
4
Enter the elements of vect_B:
5
6
7
8
The inner product of these two vectors is 70
```

Output Run 2 - Different Sizes

```
Enter the size of vectors vect_A and vect_B:
3 4
Warning!! The size of the two vectors is not the same.
```