

Data Structures and Algorithms - Activity 1

Aamn Pathak¹, 22051662, CSE-54

September 16, 2023

¹School of Computer Engineering, KIIT-DU

Question 1 Write a C function to generate all permutations of an array of distinct integers. Avoid duplicate permutations and ensure that the output includes all possible arrangements of the elements.

Code

```
1  #include <stdio.h>
2
3  void swap (int* a62, int* b62) {
4      int t = *a62;
5      *a62 = *b62;
6      *b62 = t;
7  }
8
9  void gen_perms (int arr62[], int start62, int end62) {
10     if (start62 == end62) {
11         for (int i62 = 0; i62 <= end62; i62++) {
12             printf("%d ", arr62[i62]);
13         }
14         printf("\n");
15     } else {
16         for (int i62 = start62; i62 <= end62; i62++) {
17             swap(&arr62[start62], &arr62[i62]);
18             gen_perms(arr62, start62 + 1, end62);
19             swap(&arr62[start62], &arr62[i62]);
20         }
21     }
22 }
23
24 void main () {
25
26     int n62;
27     printf("Enter the number of Elements: \n");
28     scanf("%d", &n62);
29
30     int arr62[n62];
31
32     printf("Enter the Elements: \n");
33     for (int i62 = 0; i62 < n62; i62++) {
34         scanf("%d", &arr62[i62]);
35     }
36
37     printf("The permutations are:\n");
38     gen_perms(arr62, 0, n62 - 1);
39 }
40
```

Output

```
1  dsa\assignment\1 via C v6.3.0-gcc
2  $ gcc 1.c -o 1 && ./1
3  Enter the number of Elements:
4  4
5  Enter the Elements:
6  1 3 5 7
7  The permutations are:
8  1 3 5 7
9  1 3 7 5
```

```

10      1 5 3 7
11      1 5 7 3
12      1 7 5 3
13      1 7 3 5
14      3 1 5 7
15      3 1 7 5
16      3 5 1 7
17      3 5 7 1
18      3 7 5 1
19      3 7 1 5
20      5 3 1 7
21      5 3 7 1
22      5 1 3 7
23      5 1 7 3
24      5 7 1 3
25      5 7 3 1
26      7 3 5 1
27      7 3 1 5
28      7 5 3 1
29      7 5 1 3
30      7 1 5 3
31      7 1 3 5
32
33      dsa\assignment\1 via C v6.3.0-gcc took 6s
34      $ gcc 1.c -o 1 && ./1
35      Enter the number of Elements:
36      3
37      Enter the Elements:
38      1 2 3
39      The permutations are:
40      1 2 3
41      1 3 2
42      2 1 3
43      2 3 1
44      3 2 1
45      3 1 2
46
47      dsa\assignment\1 via C v6.3.0-gcc took 6s
48      $ gcc 1.c -o 1 && ./1
49      Enter the number of Elements:
50      2
51      Enter the Elements:
52      1 5
53      The permutations are:
54      1 5
55      5 1
56

```

Question 2 Write a C program that takes two arrays as input and merges them into a third array, removing any duplicate elements while maintaining the original order.

Code

```

1  #include <stdio.h>
2
3  void merge(int arr1_62[], int arr2_62[], int m62, int n62, int
merged[], int *finalSize_62) {
4      int i62, j62;
5      *finalSize_62 = 0;
6
7      for (i62 = 0; i62 < m62; i62++) {
8          int isDuplicate_62 = 0;
9          for (j62 = 0; j62 < *finalSize_62; j62++) {
10             if (arr1_62[i62] == merged[j62]) {
11                 isDuplicate_62 = 1;
12                 break;
13             }
14         }
15         if (!isDuplicate_62) {
16             merged[(*finalSize_62)++] = arr1_62[i62];
17         }
18     }
19
20     for (i62 = 0; i62 < n62; i62++) {
21         int isDuplicate_62 = 0;
22         for (j62 = 0; j62 < *finalSize_62; j62++) {
23             if (arr2_62[i62] == merged[j62]) {
24                 isDuplicate_62 = 1;
25                 break;
26             }
27         }
28         if (!isDuplicate_62) {
29             merged[(*finalSize_62)++] = arr2_62[i62];
30         }
31     }
32 }
33 int main() {
34
35
36     int m62;
37     printf("Enter the number of Elements in array 1: \n");
38     scanf("%d", &m62);
39
40     int arr1_62[m62];
41
42     printf("Enter the Elements: \n");
43     for (int i62 = 0; i62 < m62; i62++) {
44         scanf("%d", &arr1_62[i62]);
45     }
46
47     int n62;
48     printf("Enter the number of Elements in array 2: \n");
49     scanf("%d", &n62);
50

```

```

51     int arr2_62[n62];
52
53     printf("Enter the Elements: \n");
54     for (int i62 = 0; i62 < n62; i62++) {
55         scanf("%d", &arr2_62[i62]);
56     }
57
58     int merged[m62 + n62];
59     int finalSize_62 = 0;
60
61     merge(arr1_62, arr2_62, m62, n62, merged, &finalSize_62);
62
63     printf("Final:\n");
64     for (int i62 = 0; i62 < finalSize_62; i62++) {
65         printf("%d ", merged[i62]);
66     }
67
68     return 0;
69 }
70

```

Output

```

1
2     dsa\assignment\1 via C v6.3.0-gcc
3     $ gcc 2.c
4
5     dsa\assignment\1 via C v6.3.0-gcc
6     $ ./a.exe
7     Enter the number of Elements in array 1:
8     3
9     Enter the Elements:
10    4 6 98
11    Enter the number of Elements in array 2:
12    5
13    Enter the Elements:
14    11 44 65 45 32
15    Final:
16    4 6 98 11 44 65 45 32
17    dsa\assignment\1 via C v6.3.0-gcc took 19s
18    $ ./a.exe
19    Enter the number of Elements in array 1:
20    3
21    Enter the Elements:
22    1 2 3
23    Enter the number of Elements in array 2:
24    5
25    Enter the Elements:
26    1 2 3 4 5
27    Final:
28    1 2 3 4 5
29    dsa\assignment\1 via C v6.3.0-gcc took 24s
30    $ ./a.exe
31    Enter the number of Elements in array 1:
32    5
33    Enter the Elements:
34    1 2 3 4 5
35    Enter the number of Elements in array 2:

```

```
36      3
37      Enter the Elements:
38      1 3 4
39      Final:
40      1 2 3 4 5
```

Disclaimer All the programs were compiled on `gcc.exe` (MinGW.org GCC-6.3.0-1) 6.3.0. This document was generated using \LaTeX .