Data Structures and Algorithms - Activity 1

Aamn Pathak[1], 22051662, CSE-54

September 16, 2023

[1]School 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

```c
#include <stdio.h>

void swap (int* a62, int* b62) {
  int t = *a62;
  *a62 = *b62;
  *b62 = t;
}

void gen_perms (int arr62[], int start62, int end62) {
  if (start62 == end62) {
    for (int i62 = 0; i62 <= end62; i62++) {
      printf("%d ", arr62[i62]);
    }
    printf("\n");
  } else {
    for (int i62 = start62; i62 <= end62; i62++) {
      swap(&arr62[start62], &arr62[i62]);
      gen_perms(arr62, start62 + 1, end62);
      swap(&arr62[start62], &arr62[i62]);
    }
  }
}

void main () {

  int n62;
  printf("Enter the number of Elements: \n");
  scanf("%d", &n62);

  int arr62[n62];

  printf("Enter the Elements: \n");
  for (int i62 = 0; i62 < n62; i62++) {
    scanf("%d", &arr62[i62]);
  }

  printf("The permutations are:\n");
  gen_perms(arr62, 0, n62 - 1);
}

```

Output

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

```
1 5 3 7
1 5 7 3
1 7 5 3
1 7 3 5
3 1 5 7
3 1 7 5
3 5 1 7
3 5 7 1
3 7 5 1
3 7 1 5
5 3 1 7
5 3 7 1
5 1 3 7
5 1 7 3
5 7 1 3
5 7 3 1
7 3 5 1
7 3 1 5
7 5 3 1
7 5 1 3
7 1 5 3
7 1 3 5

dsa\assignment\1 via C v6.3.0-gcc took 6s
$ gcc 1.c -o 1 && ./1
Enter the number of Elements:
3
Enter the Elements:
1 2 3
The permutations are:
1 2 3
1 3 2
2 1 3
2 3 1
3 2 1
3 1 2

dsa\assignment\1 via C v6.3.0-gcc took 6s
$ gcc 1.c -o 1 && ./1
Enter the number of Elements:
2
Enter the Elements:
1 5
The permutations are:
1 5
5 1

```

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

```
#include <stdio.h>

void merge(int arr1_62[], int arr2_62[], int m62, int n62, int
merged[], int *finalSize_62) {
  int i62, j62;
  *finalSize_62 = 0;

  for (i62 = 0; i62 < m62; i62++) {
    int isDuplicate_62 = 0;
    for (j62 = 0; j62 < *finalSize_62; j62++) {
      if (arr1_62[i62] == merged[j62]) {
        isDuplicate_62 = 1;
        break;
      }
    }
    if (!isDuplicate_62) {
      merged[(*finalSize_62)++] = arr1_62[i62];
    }
  }

  for (i62 = 0; i62 < n62; i62++) {
    int isDuplicate_62 = 0;
    for (j62 = 0; j62 < *finalSize_62; j62++) {
      if (arr2_62[i62] == merged[j62]) {
        isDuplicate_62 = 1;
        break;
      }
    }
    if (!isDuplicate_62) {
      merged[(*finalSize_62)++] = arr2_62[i62];
    }
  }
}
int main() {


  int m62;
  printf("Enter the number of Elements in array 1: \n");
  scanf("%d", &m62);

  int arr1_62[m62];

  printf("Enter the Elements: \n");
  for (int i62 = 0; i62 < m62; i62++) {
    scanf("%d", &arr1_62[i62]);
  }

  int n62;
  printf("Enter the number of Elements in array 2: \n");
  scanf("%d", &n62);

```

```c
    int arr2_62[n62];

    printf("Enter the Elements: \n");
    for (int i62 = 0; i62 < n62; i62++) {
      scanf("%d", &arr2_62[i62]);
    }

    int merged[m62 + n62];
    int finalSize_62 = 0;

    merge(arr1_62, arr2_62, m62, n62, merged, &finalSize_62);

    printf("Final:\n");
    for (int i62 = 0; i62 < finalSize_62; i62++) {
      printf("%d ", merged[i62]);
    }

    return 0;
  }

```

Output

```

dsa\assignment\1 via C v6.3.0-gcc
$ gcc 2.c

dsa\assignment\1 via C v6.3.0-gcc
$ ./a.exe
Enter the number of Elements in array 1:
3
Enter the Elements:
4 6 98
Enter the number of Elements in array 2:
5
Enter the Elements:
11 44 65 45 32
Final:
4 6 98 11 44 65 45 32
dsa\assignment\1 via C v6.3.0-gcc took 19s
$ ./a.exe
Enter the number of Elements in array 1:
3
Enter the Elements:
1 2 3
Enter the number of Elements in array 2:
5
Enter the Elements:
1 2 3 4 5
Final:
1 2 3 4 5
dsa\assignment\1 via C v6.3.0-gcc took 24s
$ ./a.exe
Enter the number of Elements in array 1:
5
Enter the Elements:
1 2 3 4 5
Enter the number of Elements in array 2:
3
Enter the Elements:
1 3 4
Final:
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.