6.9 PERMUTATIONS OF AN ARRAY

Question:

Given an array nums of distinct integers, return all the possible permutations. You can return the answer in any order.

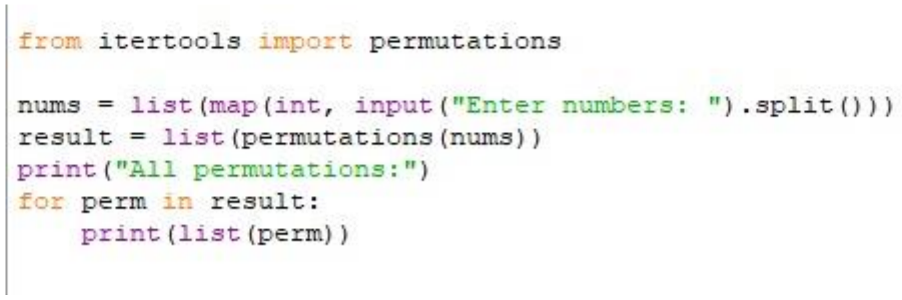
AIM

To generate all possible permutations of a given array of distinct integers using backtracking.

ALGORITHM

1. Use backtracking to generate permutations.
2. Maintain a temporary path list to build each permutation.
3. At each step, iterate over the array elements.
4. If the element is not already in the path, add it and recurse.
5. When the path length equals the length of nums, record the permutation.
6. Backtrack by removing the last added element and continue exploring.

PROGRAM



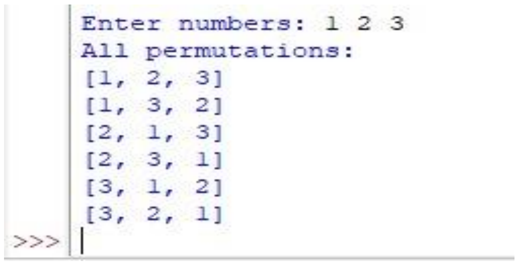
```
from itertools import permutations

nums = list(map(int, input("Enter numbers: ").split()))
result = list(permutations(nums))
print("All permutations:")
for perm in result:
    print(list(perm))
```

Input:

nums = [1,2,3]

Output:



```
Enter numbers: 1 2 3
All permutations:
[1, 2, 3]
[1, 3, 2]
[2, 1, 3]
[2, 3, 1]
[3, 1, 2]
[3, 2, 1]
>>>
```

RESULT:

Thus, the program is successfully executed and verified to generate all possible permutations of the array.

PERFORMANCE ANALYSIS:

Time Complexity: O(n * n!) where n is the number of elements, since there are n! permutations and each takes O(n) to build.

Space Complexity: O(n) for recursion depth and temporary path storage.