ADA LAB PROGRAM 3

AIM: Implement Johnson Trotter algorithm to generate permutations.

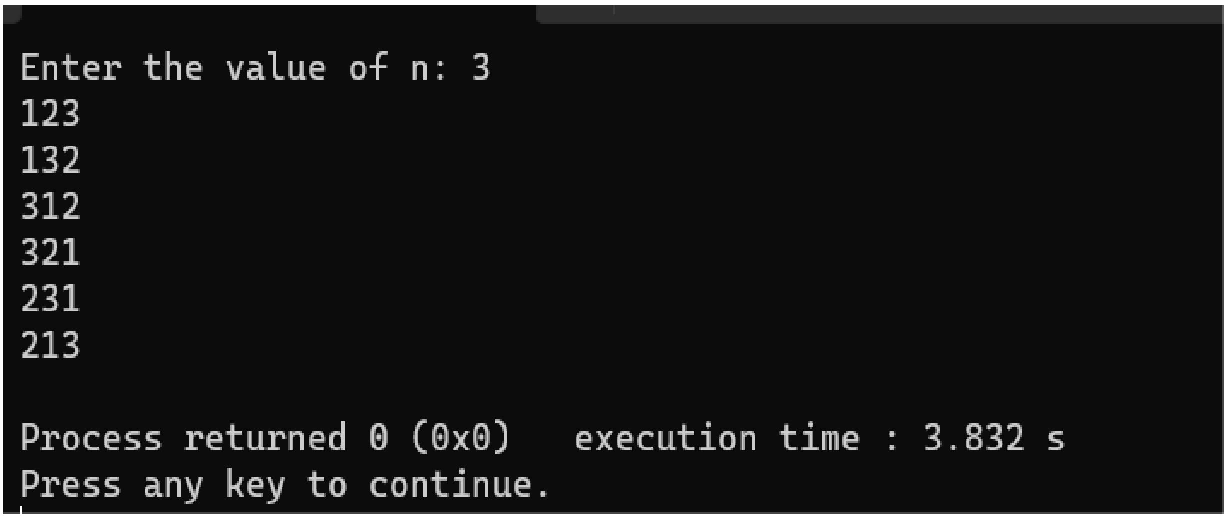
SOURCE CODE

```
#include <stdio.h>
#include <stdbool.h>
#define MAX_N 10
void swap(int *a, int *b)
{
  int temp = *a;
  *a = *b;
  *b = temp;
}
void printPermutation(int permutation[], int direction[], int n)
{
  for (int i = 0; i < n; i++)
  {
    printf("%d", permutation[i]);
  }
  printf("\n");
}
void generatePermutations(int n)
{
  int permutation[MAX_N];
  int direction[MAX_N];
  bool mobile[MAX_N];

  for (int i = 0; i < n; i++)
  {
    permutation[i] = i + 1;
    direction[i] = -1;
```

```
        mobile[i] = true;
 }
 printPermutation(permutation, direction, n);
 int mobileElement, mobileIndex, temp;
 while (true)
 {
   mobileElement = -1;
   mobileIndex = -1;
   for (int i = 0; i < n; i++)
   {
     if (direction[i] == -1 && i > 0 && permutation[i] > permutation[i - 1] && mobile[i])
     {
       if (mobileElement == -1 || permutation[i] > mobileElement)
       {
         mobileElement = permutation[i];
         mobileIndex = i;
       }
     }
     if (direction[i] == 1 && i < n - 1 && permutation[i] > permutation[i + 1] && mobile[i])
     {
       if (mobileElement == -1 || permutation[i] > mobileElement)
       {
         mobileElement = permutation[i];
         mobileIndex = i;
       }
     }
   }

   if (mobileIndex == -1)
   {
     break;
   }
```

```
        if (direction[mobileIndex] == -1)
        {
            swap(&permutation[mobileIndex], &permutation[mobileIndex - 1]);
            swap(&direction[mobileIndex], &direction[mobileIndex - 1]);
        }
        else
        {
            swap(&permutation[mobileIndex], &permutation[mobileIndex + 1]);
            swap(&direction[mobileIndex], &direction[mobileIndex + 1]);
        }
        for (int i = 0; i < n; i++)
        {
            if (permutation[i] > mobileElement)
            {
                direction[i] *= -1;
            }
        }
        printPermutation(permutation, direction, n);
    }
}
int main()
{
    int n;
    printf("Enter the value of n: ");
    scanf("%d", &n);
    if (n < 1 || n > MAX_N)
    {
        printf("Invalid input!\n");
        return 0;
    }
    generatePermutations(n);
    return 0;}
```

OUTPUT SCREENSHOT



```
Enter the value of n: 3
123
132
312
321
231
213

Process returned 0 (0x0)   execution time : 3.832 s
Press any key to continue.
```