Implement Johson Trotter to implement permutations

```
# include <stdio.h>
# include <stdlib.h>
int flag=0;
int swap (int* a, int * b)
{ int t =*a;
  * a =*b;
  *b = t;
}
int search (int arr [], int num, int mobile
{
  int g;
  for (g=0; g< num; g++)
  { if (arr [g] == mobile)
    return g+1;
  else
  flag ++;
  }
  return -1;
}
int find mobile (int arr [], int d[], int num
{
  int mobile =0;
  int mobile_p =0;
int i;
for (i=0; i<num; i++)
  {
  if (d [arr[i] -1] ==0) && i!=0)
  {
```

```
if (arr[i] > arr[i-1] && arr(i) > mobile_p)
{
  mobile = arr(i);
   mobile_p = mobile;
}
else
flag++
}
else if ((d[arr[i]-1] == 1) && i != num-1))
{
if (arr(i) > arr(i+1) && arr[i] > mobile_p)
{
 mobile = arr(i);
 mobile_p = mobile;
}
else
flag++;
}
else
flag++;
}
if (mobile_p == 0) && (mobile == 0))
return 0;
else
return mobile;
}
void permutations (int arr[], int d[],
                         int num)
{
 int i;
 int mobile = find_Mobile (arr, d, num);
 int pos = search (arr, num, mobile);
```

```
if (d[arr[pos-1]-1] ==0)
swap (&arr[pos-1], &arr[pos-2]);
else
swap (&arr[pos-1], &arr[pos]);
 for ( int i =0; i< num; i++)
{
   if (arr(i) > mobile )
{
  if  (d[arr [i]-1 ==0)
  d [arr [i]-1] =1;
  else
  d [arr [i] -1] =0;
}
}
for (i =0; i< num; i++)
{
  printf ("%d", arr [i]);
}
}
int factorial (int k)
{
 int f =1;
 int i =0;
 for (i =1; i<k+1; i++)
 f = f*i;
 return f;
}
int main ()
{
int num =20;
int [illegible]
int [illegible]
```

SPARK

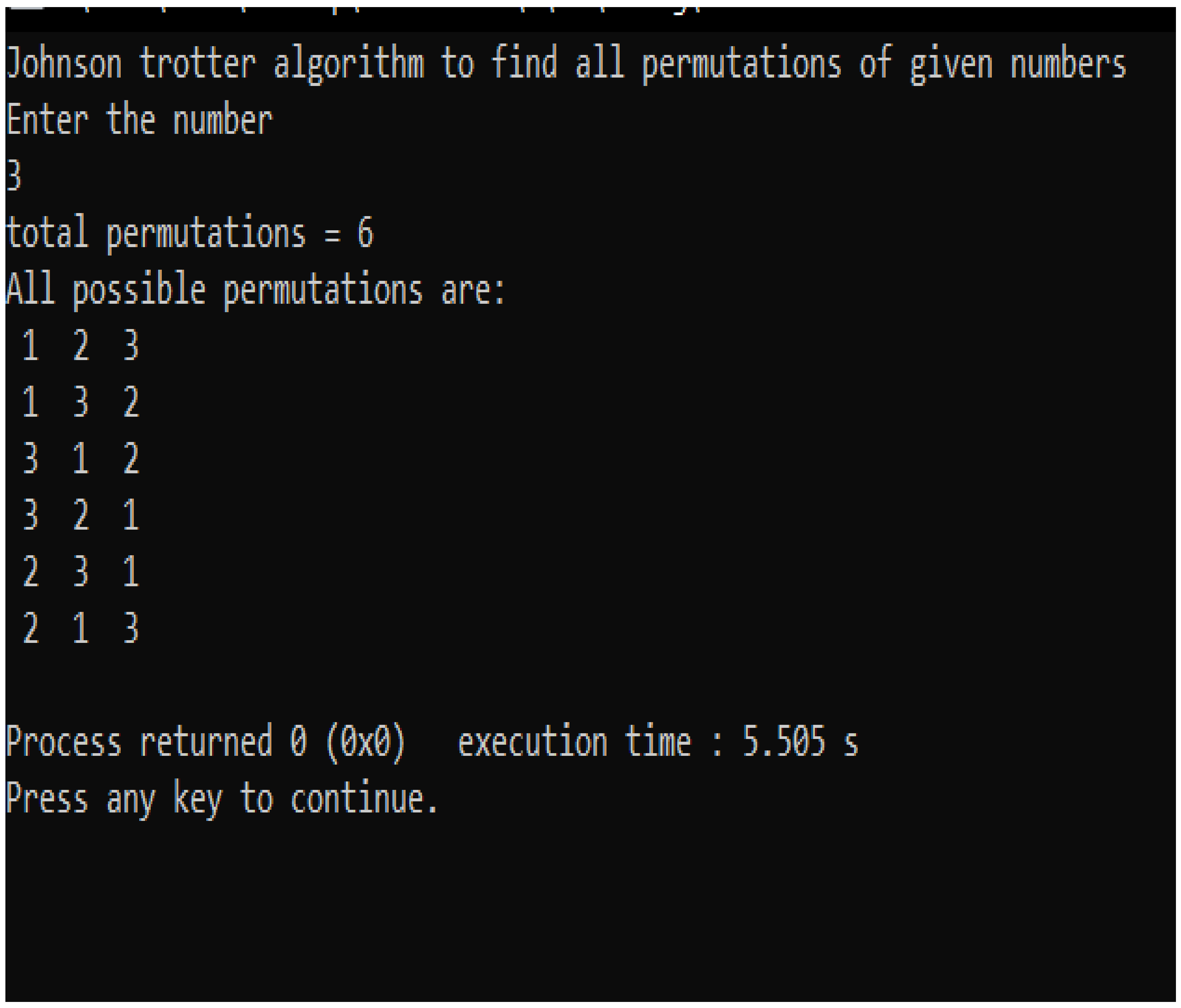
Date______ Page______

```
int z=0;
printf ("Johson Trotter algorithm to find all
            permutations)
printf ("Enter the number);
~~printf~~ (Scanf ("%d", &num);
int arr [num], d [num];
 z = factorial (num);
printf ("Total permutations = %d", z);
printf ("All possible permutations are");
 for (i=0; i< num; i++)
  {
     d [i] =0;
     arr (i) = i+1;
     printf ("%d", arr [i]);
 }
 printf ("\n");
  for (j=1; j <z; j++)
   { permutations (arr, d, num);
     print ("\n");
   }
 return 0;
}
```

Output

```
Enter the number
3
Total permutations = 6
All possible permutations are :
1  2  3
1  3  2
3  1  2
3  2  1
2  3  1
2  1  3
```

[illegible]



```
Johnson trotter algorithm to find all permutations of given numbers
Enter the number
3
total permutations = 6
All possible permutations are:
 1  2  3
 1  3  2
 3  1  2
 3  2  1
 2  3  1
 2  1  3

Process returned 0 (0x0)   execution time : 5.505 s
Press any key to continue.
```