

13/4/24

Lab-7

① Write a program - Johnson Turtles - program to find permutations of given numbers.

```
#include <stdio.h>
#include <stdlib.h>
int flag = 0;
void 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) ||
            (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 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]);
    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 (int i=0; i<num; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");
}

```



```

int factorial(int k){
    int f = 1;
    for (int i = 1; i < k + 1; i++)
        f = f * i;
    return f;
}

int main() {
    int num = 0;
    printf("Johnson Trotter algorithm to find all permutations of given numbers: \n");
    printf("Enter the number: \n");
    scanf("%d", &num);
    int arr[num], d[num];
    int z = factorial(num);
    printf("Total permutations = %d\n", z);
    printf("All possible permutations are: \n");
    for (int i = 0; i < num; i++) {
        d[i] = 0;
        arr[i] = i + 1;
        printf("%d ", arr[i]);
    }
    printf("\n");
    for (int j = 1; j < z; j++) {
        permutations(arr, d, num);
    }
    return 0;
}

```

Output:-

Johnson Trotter algorithm to find all permutations of given numbers
Enter the number: 3

Total permutations = 6

All possible permutations are:

1 2 3	3 2 1
1 3 2	2 3 1
3 1 2	2 1 3

② Pattern matching

```
#include <stdio.h>
#include <string.h>
int string_m(char t[], char p[]) {
    int n = strlen(t);
    int m = strlen(p);
    for (int i = 0; i <= (n - m); i++) {
        int j = 0;
        while (j < m && t[i + j] == p[j]) {
            j++;
        }
        if (j == m)
            return i;
    }
    return -1;
}

int main() {
    char t[100], p[100];
    printf("Enter the text: ");
    scanf("%s", t);
    printf("Enter the pattern: ");
    scanf("%s", p);
    int result = string_m(t, p);
    if (result != -1) {
        printf("Pattern found at index %d\n", result + 1);
    } else {
        printf("Pattern not found\n");
    }
    return 0;
}
```

Output:- Enter the text: helloworld
Enter the pattern: world
Pattern found at index 6