

1. Write a program to generate permutations**Source Code:**

```

#include <stdio.h>
#include <string.h>
#include <stdlib.h>

void swap(char *x, char *y)
{
    char temp;
    temp = *x;
    *x = *y;
    *y = temp;
}

int fact(int n){
    if(n == 0 )        return 1;
    else                return n * fact(n - 1);
}

void permute(char *arr, int left, int right)
{
    int i;
    static int j = 0;
    if (left == right-1)
        printf("%d - [%s]` \n", ++j, arr);
    else{
        for (i = left; i < right; i++){
            swap((arr+ left), (arr+ i));
            permute(arr, left + 1, right);
            swap((arr+ left), (arr+ i)); //backtrack
        }
    }
}

int main()
{
    int num;
    printf("Enter the number of elements: ");

```

```

scanf("%d", &num);

char *str1 = malloc(num * sizeof(char *));

printf("Enter the string: ");
for (int i = 0; i < num; i++)
    scanf("%s", &str1[i]);
printf("\n\nThe permutations in lexicographic order are:\n");
permute(str1, 0, num );
int k = fact(num);
printf("\n*****\n\n");
printf("%d permutations can be generated .\n", k);
return 0;
}

```

Output:

```

→ Lab 5 git:(master) X gcc src/01.c -o bin/01 -Wall
→ Lab 5 git:(master) X ./bin/01
Enter the number of elements: 3
Enter the string: a b c

The permutations in lexicographic order are:
1 - [ a b c ]
2 - [ a c b ]
3 - [ b a c ]
4 - [ b c a ]
5 - [ c b a ]
6 - [ c a b ]
-----
6 permutations can be generated .
→ Lab 5 git:(master) X

```

2. Write a program to generate combinationSource Code:

```

#include <stdio.h>
#include <string.h>
#include <stdlib.h>

int fact(int n){
    if(n == 0 )        return 1;
    else                return n * fact(n - 1);
}

void combinationUtil(char* arr, char *data, int start, int end, int index, int r){
    if (index == r){
        static int k = 0;
        printf("%d - [ ", ++k);
        for (int j = 0; j < r; j++)
            printf("%c ", *(data + j));
        printf("]\n");
    }
    for (int i=start; i<=end && end-i+1 >= r-index; i++){
        data[index] = arr[i];
        combinationUtil(arr, data, i+1, end, index+1, r);
    }
}

void printCombination(char arr[], int n, int r){
    char data[r];
    combinationUtil(arr, data, 0, n-1, 0, r);
}

// Driver program to test above functions
int main()
{
    int r,num;
    printf("Enter the number of elements of the array: ");
    scanf("%d", &num);

```

```

char *str =(char *) malloc(num *sizeof(char *));
printf("Enter the elements of array: ");
for (int i = 0; i < num; i++)
    scanf("%s", (str + i));

printf("Enter the number of elements to select: ");
scanf("%d", &r);

int combination;
combination = fact(num) / (fact(r)*fact(num - r));

printf("\nThe %d-combination of a set with %d distinct elements is %d and\n\
all the combination are:\n",r,num, combination);
printCombination(str, num, r);
}

```

Output:

```

→ Lab 5 git:(master) X gcc src/02.c -o bin/02 -Wall
→ Lab 5 git:(master) X ./bin/02
Enter the number of elements of the array: 5
Enter the elements of array: a b c d e
Enter the number of elements to select: 3

The 3-combination of a set with 5 distinct elements is 10 and
all the combination are:
1 - [ a b c ]
2 - [ a b d ]
3 - [ a b e ]
4 - [ a c d ]
5 - [ a c e ]
6 - [ a d e ]
7 - [ b c d ]
8 - [ b c e ]
9 - [ b d e ]
10 - [ c d e ]
→ Lab 5 git:(master) X |

```