Definicija:

Kombinacija k-te klase (bez ponavljanja) skupa An je svaki njegov podskup koji sadrži k elemenata.

Naš zadatak je da generišemo sve moguće kombinacije datog skupa elemenata. U ovom slučaju, za primer skupa brojeva uzimamo brojeve 1,2,3... i tako redom do n. Kombinacije će biti generisane u leksikografskom poretku.
Kako ovo zapravo funkcioniše?

Source code

```
#include <stdio.h>

int sledeca (int komb[], int n, int k)
{
   int l=k-1,x;/*niz pocinje od 0, zato uvodimo l*/
   while (l>=0 && komb[l]==--n) l--;/*pomeramo se u levo, ako je broj jednak n */
   if (l<0) return 0;/*slucaj greske,tj zadnja kombinacija*/
   x=komb[l];/*uzimamo vrednost poslednjeg elementa neke klase koju cemo uvecavati, tj menjati*/
   do komb[l++]=++x; while (l<k);/**/
   return 1;
}

void ispiskomb(int skup[],int k,int n)
{
   int j;
   for (j=0; j<k; j++) skup[j]=j;/*unosimo u niz 1,2,3,4,5,6....*/
   do
   { for (j=0; j<k; j++) printf("%d\t", skup[j]+1); /*stampa pojedinacni clan niza uvecavajuci ga za 1 jer niz u
C-u pocinje od 0*/
     printf("\n");
   }
   while(sledeca(skup, n, k));
}

void kombinacije(){
int n,k,skup[20];
printf("Unesite klasu k i broj elemenata n:\n");
scanf("%d %d",&k,&n);
ispiskomb(skup,k,n);
}

int main(){
kombinacije();
return 0;}

/*Primer izlaza:
Unesite klasu k i broj elemenata n:
3
5
1       2       3
1       2       4
1       2       5
1       3       4
1       3       5
1       4       5
2       3       4
2       3       5
2       4       5
3       4       5 */
```

Krećemo se na sledeći način:

Broj koji posmatramo se poredi sa brojem članova, u slučaju da je taj broj manji od n (broj članova), broj uvećevamo za jedan i time dobijamo novu kombinaciju koju štampamo i tako nastavljamo sve do slučaja kada je taj broj jednak broju n. Nakon toga *l* (možemo ga smatrati kao neki pokazivač na elemente jedne kombinacije) se pomera, tj u datom slucaju na 2. član, u levo i njegova vrednost uvećava za jedan i onda vrednost tog broja uvećavamo za jedan pri svakom prolazu i dodeljujemo sledećem članu sve dok je taj član manji ili jednak od n. Kada nastupi taj slučaj, ponovo uvećavamo vrednost onog elementa na koji *l* pokazuje i ponavljamo postupak sve dok je uvećani posmatrani broj manji ili jednak od n kada *l* prelazi na sledeći element ulevo. Postupak se ponavlja do poslednje kombinacije.