**Recursion**

* Solving Return all codes
* Solving print all codes
* Merge Sort
* Print pascal triangle using one recursion function
* **package** lecture13;
* **import** java.util.Scanner;
* **public** **class** PatterUsingRecursion {
* **public** **static** **void** print(**int** row,**int** k,**int** n,**int** mark){
* **if**(row>n)
* **return** ;
* **if**(k<=(n-row) && mark==0)
* {
* System.***out***.print(" ");
* *print*(row,++k,n,mark);
* }
* **else** **if**(k<=row && mark==1)
* {
* **if**(row==1)
* {
* System.***out***.println(" "+1);
* *print*(++row,0,n,0);
* }
* **else**{
* System.***out***.print(*permutation*(row,k)+" ");
* *print*(row,++k,n,mark);
* }
* }
* **else**{
* **if**(mark==0)
* *print*(row,0,n,1);
* **else**
* {
* System.***out***.println();
* *print*(++row,0,n,0);
* }
* }
* **return** ;
* }
* **public** **static** **int** permutation(**int** n,**int** r){
* **int** k=*fact*(n)/(*fact*(r)\**fact*(n-r));
* **return** k;
* }
* **public** **static** **int** fact(**int** n){
* **int** k=1;
* **while**(n>0)
* {
* k\*=n--;
* }
* **return** k;
* }
* **public** **static** **void** main(String argsp[]) {
* // **TODO** Auto-generated method stub
* Scanner sc= **new** Scanner(System.***in***);
* **int** n = sc.nextInt();
* *print*(1,1,n,0);
* sc.close();
* }
* }