if (n <= 4)
return 1;

Ques!

1	2	3	4	5	6	7	8	9
1	1	1	1	4	7	13	25	49

Faith (n-1)

If we give (n-1) (n-2), it will return the right ans

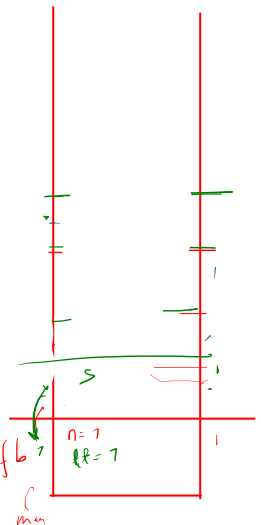
```
public static int fb(int n){
    int lt = fb(n-1);
    int lst = fb(n-2);
    int ltt = fb(n-3);
    int lft = fb(n-4);
    int ans = lt + lst + ltt + lft;
    return ans;
}
```

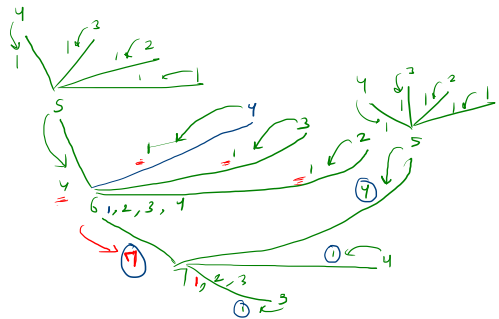
Expectation

It should return number of players in nth round.

n = 7, ans = 13

f(n) = f(n-1) + f(n-2) + f(n-3) + f(n-4)





```
public static int fourbinacci(int n) {
    int lt = fourbinacci(n-1);
    int lst = fourbinacci(n-2);
    int ltt = fourbinacci(n-3);
    int lft = fourbinacci(n-4);

    int ans = lt + lst + ltt + lft;
    return ans;
}
```

if (n <= 4)
return 1

7 + 4 + 1 + 1 => 13

Class 2

Josephus problem

n = 6, k = 5

n = 6, k = 5

ans = 1

kth person

k = 5 ⇒ k = 4

0 + 4

k = 5 ⇒ k = 4

al = 1 2 3 4 5 6

idx = 0 ⇒ (0 + 4) % al.size ⇒ 4 % 6 ⇒ 4

al = 1 2 3 4 6

idx = 4 + 4 ⇒ 8 % al.size ⇒ 8 % 5 ⇒ 3

al = 1 2 3 6

idx = (3 + 4) ⇒ 7 % al.size ⇒ 7 % 4 ⇒ 3

al = 1 2 3

idx = (3 + 4) ⇒ 7 % al.size ⇒ 7 % 3 ⇒ 1

al = 1

idx = (1 + 4) ⇒ 5 % al.size ⇒ 5 % 2 ⇒ 1

ArrayList<Integer> al = new ArrayList<>(); // size = 0

2 5 9 ⇒ 2 7

add

al.add(2) // size = 1

al.add(5)

al.add(9)

get

al.get(1) // al.get(idx)

remove

al.remove(1)

size

al.size();

n = 6; k = 5 k = 4

idx = 0

al ⇒ 1 2 3 4 5 6

ptk = (0 + 4) % 6 ⇒ 4 % 6 ⇒ 4

al = 1 2 3 4 6

ptk = (4 + 4) % 5 ⇒ 8 % 5 ⇒ 3

idx = 4

idx = 3

al ⇒ 1 2 3 6

ptk ⇒ (3 + 4) % 4 ⇒ 7 % 4 ⇒ 3

idx = 3

al ⇒ 1 2 3

ptk = (3 + 4) % 3 ⇒ 7 % 3 ⇒ 1

idx = 1

al ⇒ 1 3

ptk ⇒ (1 + 4) % 2 ⇒ 5 % 2 ⇒ 1

idx = 1

al ⇒ 1

```java
public static int deathGame(int n, int k) {
    ArrayList<Integer> al = new ArrayList<>();

    for(int i=1; i<=n; i++){
        al.add(i);
    }

    k--;
    int idx=0;
    while(al.size()>1){
        int personToKill = (idx+k)%al.size();
        al.remove(personToKill);

        idx = personToKill;
    }

    return al.get(index: 0);
}
```


```java
public static boolean isLuckyNumber(int num){
    if(num==1){
        return true;
    }
    if(num==89){
        return false;
    }

    int sumOfSquareDigits = 0;

    while(num>0){
        int rem = num%10;
        sumOfSquareDigits += rem*rem;

        num/=10;
    }

    return isLuckyNumber(sumOfSquareDigits);
}
```

```java
static int solve(int n) {
   for(int i=n+1; i<=10000; i++){
        boolean luckyNumber = isLuckyNumber(i);
        if(luckyNumber == true){
            return i;
        }
   }

   return -1;
}
```