- Now the problem is you have to compute nCr % P, where n,r<=10^18 but P<=10^6.
- We need to solve this problem using Lucas theorem. Because in Lucas theorem problem will be reduced to sub problems. In this theorem the n and r are converted to P base number and then we compute the same digit-location wise binomial coefficients. Lucas theorem is given below :

$$\binom{n}{r} \equiv \prod_{i=0}^{k} \binom{n_i}{r_i} \pmod P$$

Where n $=(n_k \ldots \ldots\ n_2\ n_1\ n_0\ )_P$, r $=(r_k \ldots \ldots\ r_2\ r_1\ r_0\ )_P$

```
long long in[1000001], fact[1000001], ifact[1000001];

void generate(long long MX, long long P)
{
          fact[0]=1;
          for(int i=1;i<=MX;i++) fact[i]=(1LL * fact[i-1] * i)%P;
          in[0]=0,in[1]=1;
          for(int i=2;i<=MX;i++) in[i]= (1LL * ( (P-1)* (P/i) )%P * in[P%i] )%P;
          ifact[0]=1;
          for(int i=1;i<=MX;i++) ifact[i]=(1LL * ifact[i-1] * in[i] )%P;
}

long long small_nCr(long long n, long long r, long long P)
{
         if(r>n)return 0;

         long long ans;
         ans=(1LL * fact[n] * ifact[n-r])%P;
         ans=( ans * ifact[r] )%P;

         return ans;
}

long long nCr(long long n, long long r, long long P)
{
  if(r==0)return 1;

  long long ni=(n%P), ri=(r%P);

  return ( nCr(n/P, r/P, P)* small_nCr(ni, ri, P)) %P;
}

long long Lucas(long long n, long long r, long long P)
{
         generate(P, P);
         return nCr(n, r, P);
}
```

```
int main()
{

    long long prime[]={13,29,67,113,157,223};
    for(int i=0;i<6;i++)
    {
        cout<<( 126%prime[i] )<<'  '<<Lucas(9, 5, prime[i] )<<endl;
    }
}
```

- Now the problem is, if we want to find nCr % P, where **P is not a prime** number. Solution: We can split P into its prime divisors and count binomial coefficients for each divisor and marge them using Chinese remainder theorem.

Chinese Remainder Theorem:

P can be written as, $P = P_1 P_2 \dots \dots P_{n-1} P_n$(prime divisors)

Now we can separately calculate,

$$n_{C_r} = X$$

$$X \equiv r_1 \ (mod\ P_1)$$

$$X \equiv r_2 \ (mod\ P_2)$$

$$\dots \dots \dots$$

$$X \equiv r_n \ (mod\ P_n)$$

Now we can marge the above equations using Chinese remainder theorem. By using this theorem we can find the minimum value of X for which all the equation are true along with the below one.

$$X \equiv r \ (mod\ P)$$

X can found by using followed equation,

$$X = \sum_{i=1}^{n} (r_i * pd_i * inv(pd_i\,, P_i))$$

$$pd_i = \frac{P}{P_i} \quad \& \ r = X\%P$$

```
long long  n, prime[20], rim[20];

long long  bigMod(long long  b, long long  p, long long  M)
{
        if(p==0)return 1;

    long long  tmp= bigMod (b, p/2,M);

    tmp = (tmp * tmp)% M;

    return (p%2==0)? tmp : ( b * tmp) % M;

}

long long  INV(long long  num, long long M)
{
        return bigMod(num,M-2,M);
}



long long ChineseRemainder( )
{
     long long  product=1, x=0, pd;
     for(int i=0;i<n;i++)
          product*=prime[i];

     for(int i=0;i<n;i++)
     {

          pd=product/prime[i];

          x+=( rim[i] * pd * INV( pd, prime[i]) );

          x%=product;
     }

return x;

}
```

"Just don't laugh at my English"- Kamol :D