// A function that return the value of number of digit

// in N!

ll factorialDigit(int n )

{

double res= 0.0 ;

for(int i=1; i<=n ; i++)

{

res+= log10(i) ;

}

res = floor(res)+ 1 ;

return res ;

}

\*\*\* where n >10^6 \*\*\*

// Returns the number of digits present

// in n! Since the result can be large

// long long is used as return type

long long findDigits(int n)

{

// factorial of -ve number

// doesn't exists

if (n < 0)

return 0;

// base case

if (n <= 1)

return 1;

// Use Kamenetsky formula to calculate

// the number of digits

double x = ((n \* log10(n / M\_E) +

log10(2 \* M\_PI \* n) /

2.0));

return floor(x) + 1;

}

/\*\*\* Lightoj Digit in factorial in different base \*\*\*/

Solution ::

using namespace std ;

double ara[1000005] ;

void preCal(){

for(int i=1 ; i<=1000005 ; i++){

ara[i] = ara[i-1] + log(i) ;

}

}

int main(){

preCal() ;

int t,tcase,n,base;

scanf("%d",&t) ;

tcase =1 ;

while(t--){

scanf("%d%d",&n,&base) ;

double digit = 0.0 ;

if(n==0 || n== 1 ){

printf("Case %d: 1\n",tcase++) ;

}

else {

digit = ceil(ara[n]/log(base)) ;

//digit = (int)digit + 1 ;

printf("Case %d: %d\n",tcase++,(int)digit) ;

}

}

return 0 ;

}

Formula :: loga^x = (logb^x / logb^a)