# Контрольная работа. Функции.

Во всех задачах приведена некоторая программа, в которой вызывается одна или несколько функций.
Вам надо сдать в тестирующую систему файл, в котором определены эти функции так, чтобы приведённая программа верно решала поставленную задачу.

A. Приведена программа, считывающая два натуральных числа $A$ и $B$ и печатающая несократимую дробь, равную $\frac{A}{B}$.

```
# your function(s) here
```

```
A, B = int(input()), int(input())
x = gcd(A, B)
print(A // x, '/', B // x, sep = '')
```

| Input | Output |
|---|---|
| 345<br>3672456 | 5/53224 |

B. Известен следующий математический факт: несократимая обыкновенная дробь представляется в виде конечной десятичной дроби в том и только в том случае, когда её знаменатель в разложении на простые множители имеет только числа 2 и 5.

Приведена программа, которая выводит `YES` в случае, если дробь $\frac{N}{K}$ представляется в виде конечной десятичной дроби и выводит `NO` в противном случае.

```
# your function(s) here
```

```
N, K = int(input()), int(input())
g = gcd(N, K)
if divisible_2_5(K // g):
    print('YES')
else:
    print('NO')
```

| Input | Output |
|---|---|
| 3<br>6 | YES |
| 3<br>7 | NO |
| 13<br>160 | YES |

C. Приведена программа, относительно эффективно вычисляющая биномиальный коэффициент по известной формуле:

$$C_N^K = \frac{N!}{(N-K)!K!}$$

```
# your function(s) here
```

```
N, K = int(input()), int(input())
cnk = 1
for p in range(2, N + 1):
    if is_prime(p):
        d1 = prime_multiplicity(N, p)
        d2 = prime_multiplicity(N - K, p)
        d3 = prime_multiplicity(K, p)
        cnk *= power(p, d1 - d2 - d3)
print(cnk)
```

| Input | Output |
|---|---|
| 4<br>2 | 6 |
| 16<br>10 | 8008 |

D. Приведена программа, которая вычисляет значение арифметического выражения, введённого в виде символьной строки. Выражение содержит только целые неотрицательные числа и знаки сложения и вычитания. Первое число может быть отрицательным. Числа не превосходят $2 \cdot 10^6$, их количество не превосходит $10^5$.

```
# your function(s) here
```

```
s = input()

ans = 0
now = 0
sign = 1
if s[0] == '-':
    sign = -1
now_num, k = get_number(s, 0)
ans = now_num
while k < len(s):
    sign = s[k]
    k += 1
    now_num, k = get_number(s, k)
    if sign == '+':
        ans += now_num
    else:
        ans -= now_num

print(ans)
```

| Input | Output |
|---|---|
| 1+12-54+68-17 | 10 |
| -17-17 | -34 |

E. Ниже приведён один из возможных *наивных* алгоритмов умножения многочленов. Каждый многочлен задан массивом коэфициентов, где $x[k]$ — коэффициент при одночлене $x^k$.
В первой строке вводятся два числа: степени многочленов.
Во второй строке вводятся коэффициенты первого многочлена, в третьей строке коэффициенты второго многочлена. Коэффициенты вводятся, начиная от свободного члена, т.е. коэффициента при $x^0$.

```
# your function(s) here
```

```
x = list(map(int, input().split()))
y = list(map(int, input().split()))

res = [0] * (len(x) + len(y) - 1)
for k in range(len(y) - 1, -1, -1):
    tmp = shift(mult_const(x, y[k]), k)
    array_add(res, tmp)

print(' '.join(map(str, res)))
```

| Input | Output |
|---|---|
| 0 0<br>2<br>3 | 6 |
| 1 1<br>-2 1<br>3 1 | -6 1 1 |
| 2 3<br>-3 0 1<br>-7 2 0 -1 | 21 -6 -7 5 0 -1 |

*Комментарий ко второму тесту:* $(x-2)(x+3) = x^2 + x - 6$