**For1**. *Даны целые числа K и N  (N > 0). Вывести N раз число K.*

Организовываем простой цикл от 1 до требуемого числа.

program for1;

var

K, N, i: integer;

begin

read(K, N);

for i := 1 to N do write(K, ' '); {Пишем К через пробел }

end.

**For2**. Даны два целых числа A и B (A < B). Вывести в порядке возрастания все целые числа, расположенные между A и B (включая сами числа A и B), а также количество N этих чисел.

Так как A < B, то цикл должен будет выводить все числа от А до B. Чтобы сосчитать количество чисел, используем формулу: <конечное\_значение> — <начальное\_значение> + 1.

program for2;

var

A, B, i, count: integer;

begin

read(A, B);

for i := A to B do write(i, ' '); {выписываем числа от меньшего к большему}

count := B - A + 1; {считаем количество чисел}

writeln;

write( 'Количество чисел - ', count);

end.

**For9**. *Даны два целых числа A и B (A < B). Найти сумму квадратов всех целых чисел от A до B включительно.*

Организовываем такой же цикл, как и в предыдущей задаче, но одновременно суммируем квадраты всех чисел. Чтобы высчитать квадрат, используем функцию [Sqr](http://learnpascal.ru/vvedenie-v-paskal/matematica-v-pasca.html).

program for9;

var

A, B, i, S: integer;

begin

read(A, B);

S := 0; {PascalABC делает это автоматически, но если у вас другой компилятор советуем обнулять переменные вручную}

for i := A to B do S := S + Sqr(i); {складываем все квадраты}

writeln;

write( 'Сумма квадратов - ', S);

end.

**For13°**. *Дано целое число N (> 0). Найти значение выражения 1.1 – 1.2 + 1.3 – … (N слагаемых, знаки чередуются). Условный оператор не использовать.*

Для того, чтобы поменять знак, каждую итерацию цикла меняем значение специальной переменной на противоположное.

program for13;

var

N, A, i: integer;

S: real;

begin

Write('N = ');

readln(N);

S := 1.1;

A := 1; {Сначала положительное}

for i := 2 to N do {первую итерацию цикла мы уже произвели, поэтому начинаем отсчет с 2}

begin

A := -A; {Теперь отрицательное}

S := S + A \* (1 + i / 10); {Складываем}

end;

Writeln(S:5:1); {Отдадим под дробную часть одно знакоместо}

end.

**While1°**. *Даны положительные числа A и B (A > B). На отрезке длины A размещено максимально возможное количество отрезков длины B (без наложений). Не используя операции умножения и деления, найти длину незанятой части отрезка A.*

Каждый раз вычитаем B из А, пока А — В  >= 0.

program while1;

var

A, B: integer;

begin

readln(A,B);

while (A - B) >= 0 do A := A - B; {Пока разница положительная, вычитаем. Необходимо предусмотреть вариант с кратностью А и B, поэтому >=}

write(A);

end.

**While4°**. *Дано целое число N (> 0). Если оно является степенью числа 3, то вывести True, если не является — вывести False.*

Действуем следующим образом: пока N делится нацело на три, делим N нацело. Затем, если N = 1 — число является степенью тройки; если N <> 1, тогда число — не степень тройки. Для того чтобы решить эту задачу, требуется знать, что такое [div и   mod](http://learnpascal.ru/vvedenie-v-paskal/matematica-v-pasca.html),  и как работают [логические выражения](http://learnpascal.ru/vvedenie-v-paskal/boolean.html).

program while4;

var

N: integer;

begin

readln(N);

while N mod 3 = 0 do N := N div 3; {Пока остаток от деления на три равен нулю, делим N нацело }

writeln(N = 1); {логическое выражение}

end.