Семинар #1: Основы. Домашнее задание.

Задача 1. Условие:

Напишите программу, которая будет считывать число и проверять, является ли число чётным и принадлежащим следующему множеству $[0, 20] \cup (100, 200)$ и печатать `Yes` или `No`. Используйте один оператор `if`.

вход	выход
4	Yes
5	No
20	Yes
22	No
100	No
102	Yes
202	No

Задача 2. Три числа:

На вход программе подаются три числа: `a`, `b` и `c`. Нужно проверить следующие условия:

1. Если числа `a`, `b` и `c` являются последовательными, то нужно напечатать `Consecutive`.
2. Если последовательность `a`, `b`, `c` является возрастающей, то нужно напечатать `Increasing`.
3. Если последовательность `a`, `b`, `c` является убывающей, то нужно напечатать `Decreasing`.
4. Если все три числа равны, то нужно напечатать `Equal`.
5. В ином случае нужно напечатать `None`.

вход	выход
1 2 3	Consecutive Increasing
1 2 4	Increasing
1 1 2	None
1 2 1	None
1 5 9	Increasing
1 0 -1	Consecutive Decreasing
1 5 4	None
7 7 7	Equal
20 15 5	Decreasing

Задача 3. Число, квадрат и куб:

Напишите программу, которая будет печатать само число, его квадрат и его куб от `1` до `n`, разделённые стрелочкой. Число `n` считывается с помощью `scanf`. Например, при `n = 5`, программа должна напечатать следующее:

```
1 ->  1 ->   1
2 ->  4 ->   8
3 ->  9 ->  27
4 -> 16 ->  64
5 -> 25 -> 125
```

Для того чтобы все числа печатались выровнено, можно использовать спецификатор `%3i` за место `%i` в `printf`. В этом случае, если число имеет в записи меньше 3-х цифр, то `printf` напечатает необходимое число пробелов перед числом.

Задача 4. Последовательность:

Пример программы, которая считывает число `n`. Затем считывает `n` чисел и находит среди них максимум.

```
#include <stdio.h>
#include <limits.h>
int main()
{
    int n;
    scanf("%i", &n);
    int max = INT_MIN;
    for (int i = 0; i < n; ++i)
    {
        int a;
        scanf("%i", &a);
        if (a > max)
            max = a;
    }
    printf("Max = %i\n", max);
}
```

В этой программе используется константа `INT_MIN` из библиотеки `limits.h`. Эта константа равна минимальному возможному значению чисел типа `int`, то есть `INT_MIN = -2147483648`.

Подзадачи:

Измените программу выше так чтобы:

1. Программа находила минимум, а не максимум. Может понадобиться константа `INT_MAX = 2147483647`.
2. Программа находила минимальное чётное число и максимальное нечётное. Если чётных или нечётных чисел нет, то программа должна печатать `None` за место числа.

вход	выход
3 4 5 6	4 5
3 7 7 7	None 7
10 1 8 2 4 8 8 1 5 2 8	2 5
4 10 8 6 8	6 None

3. Программа находила максимум и количество элементов, равных этому максимуму.

вход	выход
3 1 2 3	3 1
3 7 7 7	7 3
10 1 8 2 4 8 8 1 5 2 8	8 4

4. Программа печатала `Increasing` если последовательность чисел строго возрастает, `Decreasing`, если последовательность чисел строго убывает и `Equal`, если все члены последовательности равны. В любом ином случае программа должна печатать `None`.

вход	выход
3 1 2 3	Increasing
3 7 7 7	Equal
5 20 15 10 7 5	Decreasing
4 1 1 4 5	None

Задача 5. Числа-градины I:

Пусть нам на вход поступает число `n`. Мы преобразуем это число следующим образом $n = f(n)$, где

$$f(n) = \begin{cases} 3n+1, & \text{если } n - \text{нечётное} \\ n/2, & \text{если } n - \text{чётное} \end{cases}$$

Затем повторяем этот алгоритм до тех пор пока число не достигнет единицы. Получится некоторая последовательность. Например, если изначально `n = 7`, то последовательность будет выглядеть следующим образом:

```
7 22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
```

Ваша задача заключается в том, чтобы напечатать эту последовательность, её длину и максимальный элемент этой последовательность по изначальному числу `n`.

вход	выход
3	3 10 5 16 8 4 2 1 Length = 8, Max = 16
256	256 128 64 32 16 8 4 2 1 Length = 9, Max = 256
7	7 22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1 Length = 17, Max = 52

Задача 6. Числа-градины II:

На вход поступает 2 числа `a` и `b`. Нужно найти такое число `n` ($a \le n \le b$), для которого последовательность чисел-градин будет самой длинной. Нужно напечатать число `n`, а также длину последовательности, которая начинается с `n`.

вход	выход
1 5	3 8
1 8	7 17
1 10	9 20
10 15	14 18
1 100	97 119
1 500	327 144
400 500	487 142
1 1000	871 179
1 10000	6171 261
1 100000	77031 351

Задача 7. Сумма:

На вход программе подаются два целых числа `n` и `m`. Нужно посчитать следующую сумму:

$$S_{n,m} = \sum_{i=1}^{n} \sum_{j=1}^{m} (-1)^{i+j} i \cdot j$$

Например, если `n = 3`, а `m = 4`, то сумма будет равна:

$$S_{3,4} = 1 - 2 + 3 - 4 - 2 + 4 - 6 + 8 + 3 - 6 + 9 - 12 = -4$$

вход	выход
1 1	1
2 2	1
3 3	4
3 4	-4
5 7	12
10 10	25
77 107	2106

Задача 8. Печать всех делимых:

На вход программе подаются числа `a`, `b`, `c`. Программа должна напечатать все числа, делящиеся на `c` на отрезке `[a, b]` через пробел.

вход	выход
1 20 4	4 8 12 16 20
1 20 7	7 14
1 10000 9500	9500
1 1000000000 500000000	500000000 1000000000
1 1000000000 123456789	123456789 246913578 370370367 493827156 617283945 740740734 864197523 987654312

Задача 9. Пифагоровы тройки:

На вход приходит целое число `n`. Нужно напечатать все возможные пифагоровы тройки a, b и c, такие что $a \leq n$, $b \leq n$ и $c \leq n$. Пифагорова тройка – это тройка натуральных чисел, для которых верно:

$$a^2 + b^2 = c^2$$

Пифагоровы тройки, получаемые из некоторой пифагоровой тройки путём обмена местами чисел a и b считаются дублирующими. Пифагоровы тройки, получаемые из некоторой пифагоровой тройки путём умножения всех чисел на некоторое натуральное число, также считаются дублирующими. Печатать дублирующие тройки не нужно.

Подсказка: Просто переберите все возможные значения a, b и c.

вход	выход
15	3 4 5 5 12 13
50	3 4 5 5 12 13 8 15 17 7 24 25 20 21 29 12 35 37 9 40 41