Problem One – Largest Palindrome Product

A palindromic number reads the same both ways. The largest palindrome made from the product of two 2-digit numbers is 9009 = 91 × 99.

Write a program to determine the largest palindrome made from the product of two 3-digit numbers.

Answer: 906609

Problem Two – Smallest Multiple

2520 is the smallest number that can be divided by each of the numbers from 1 to 10 without any remainder.

Write a program to compute the smallest positive number that is *evenly divisible* by all of the numbers from 1 to 20?

Answer: 232792560

Problem Three – Largest Products in a Grid

In the 20×20 grid below, four numbers along a diagonal line have been marked in red.

08 02 22 97 38 15 00 40 00 75 04 05 07 78 52 12 50 77 91 08
49 49 99 40 17 81 18 57 60 87 17 40 98 43 69 48 04 56 62 00
81 49 31 73 55 79 14 29 93 71 40 67 53 88 30 03 49 13 36 65
52 70 95 23 04 60 11 42 69 24 68 56 01 32 56 71 37 02 36 91
22 31 16 71 51 67 63 89 41 92 36 54 22 40 40 28 66 33 13 80
24 47 32 60 99 03 45 02 44 75 33 53 78 36 84 20 35 17 12 50
32 98 81 28 64 23 67 10 **26** 38 40 67 59 54 70 66 18 38 64 70
67 26 20 68 02 62 12 20 95 **63** 94 39 63 08 40 91 66 49 94 21
24 55 58 05 66 73 99 26 97 17 **78** 78 96 83 14 88 34 89 63 72
21 36 23 09 75 00 76 44 20 45 35 **14** 00 61 33 97 34 31 33 95
78 17 53 28 22 75 31 67 15 94 03 80 04 62 16 14 09 53 56 92
16 39 05 42 96 35 31 47 55 58 88 24 00 17 54 24 36 29 85 57
86 56 00 48 35 71 89 07 05 44 44 37 44 60 21 58 51 54 17 58
19 80 81 68 05 94 47 69 28 73 92 13 86 52 17 77 04 89 55 40
04 52 08 83 97 35 99 16 07 97 57 32 16 26 26 79 33 27 98 66
88 36 68 87 57 62 20 72 03 46 33 67 46 55 12 32 63 93 53 69
04 42 16 73 38 25 39 11 24 94 72 18 08 46 29 32 40 62 76 36
20 69 36 41 72 30 23 88 34 62 99 69 82 67 59 85 74 04 36 16

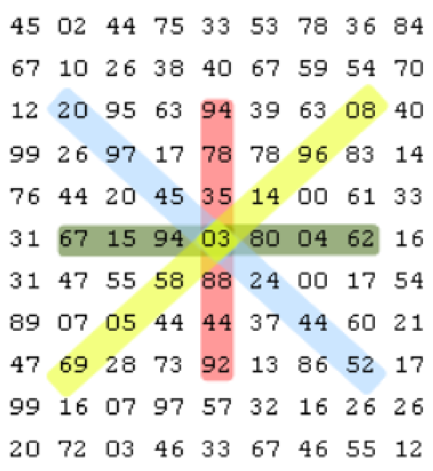
20 73 35 29 78 31 90 01 74 31 49 71 48 86 81 16 23 57 05 54
01 70 54 71 83 51 54 69 16 92 33 48 61 43 52 01 89 19 67 48

The product of these numbers is 26 × 63 × 78 × 14 = 1788696.

Write a program to determine what is the greatest product of four adjacent numbers in the same direction (up, down, left, right, or diagonally) in the 20×20 grid?

Hint:

As shown on the figure below, whenever you are looking at one number, you need to check eight directions to find the maximum product of that particular number. However, since you will visit all numbers, we will only need to check four different directions for each number, since the rest of the directions are already being checked when you vist(ed) another number.



45	02	44	75	33	53	78	36	84
67	10	26	38	40	67	59	54	70
12	20	95	63	94	39	63	08	40
99	26	97	17	78	78	96	83	14
76	44	20	45	35	14	00	61	33
31	67	15	94	03	80	04	62	16
31	47	55	58	88	24	00	17	54
89	07	05	44	44	37	44	60	21
47	69	28	73	92	13	86	52	17
99	16	07	97	57	32	16	26	26
20	72	03	46	33	67	46	55	12

For example, when you are examining the marked number 03, you don't need to check the up directions to find out that the product is 03*35*78*94 = 769860. Why? You already knew that since you would have checked the down direction when you examined 94 (three lines above).

You should aim to store the numbers in a 2D array and then process it accordingly to find your largest adjacent product.

Answer: 70600674

Problem Four – Collatz Problem

This task is based on a problem posed by Lothar Collatz in 1937. The following iterative sequence is defined for the set of positive integers:

$n \rightarrow n/2$ (n is even)
$n \rightarrow 3n + 1$ (n is odd)

Using the rule above and starting with 13, we generate the following sequence:

$$13 \rightarrow 40 \rightarrow 20 \rightarrow 10 \rightarrow 5 \rightarrow 16 \rightarrow 8 \rightarrow 4 \rightarrow 2 \rightarrow 1$$

Explained.

As 13 is odd, triple it and add 1.

N is now = 40. As 40 is even, divide it by 2.

N is now 20. As 20 is even, divide it by 2.

N is now 10. As 10 is even, divide it by 2.

N is now 5. As 5 is odd, triple it and add 1.

N is now 16. As 16 is even, divide it by 2.

N is now 8. As 8 is even, divide it by 2.

N is now 4. As 4 is even, divide it by 2.

N is now 2. As 2 is even, divide it by 2.

N is now 1.

It can be seen that this sequence (starting at 13 and finishing at 1) contains 10 terms. Although it has not been proved yet, it is thought that no matter what number you start at, you will eventually finish at 1. The following example assumes n is initially 6, and yields the following sequence.

$$6 \rightarrow 3 \rightarrow 10 \rightarrow 5 \rightarrow 16 \rightarrow 8 \rightarrow 4 \rightarrow 2 \rightarrow 1.$$

Write a program which will determine which starting number, under one million, produces the longest chain?

NOTE: Once the chain starts the terms are allowed to go above one million.

Answer: 837799

Problem Five – Counting Sundays

You are given the following information, but you may prefer to do some research for yourself.

- 1 Jan 1900 was a Monday.
- Thirty days has September,
  April, June and November.
  All the rest have thirty-one,
  Saving February alone,
  Which has twenty-eight, rain or shine.
  And on leap years, twenty-nine.
- A leap year occurs on any year evenly divisible by 4, but not on a century unless it is divisible by 400.

Write a program to determine how many Sundays fell on the 1st of the month during the twentieth century (1 Jan 1901 to 31 Dec 2000)?

Hint:

This problem can be tackled in the shortest possible time by using a combination of the GregorianCalendar and Calendar classes from the Java API.

Answer: 171

Problem Six – Prime Permutations

The arithmetic sequence, 1487, 4817, 8147, in which each of the terms increases by 3330, is unusual in two ways:

1) Each of the three terms are prime.
2) Each of the 4-digit numbers are permutations of one another.

There are no arithmetic sequences made up of three 1-, 2-, or 3-digit primes, exhibiting this property, but there is one other 4-digit increasing sequence.

What 12-digit number do you form by concatenating the three terms in this sequence? Write a program to determine this number.

Answer: 296962999629

These exercises are excellent for building on your problem solving and logical skills. The six problems are well known and the code for them is widely available online – you will learn nothing from sourcing this code.

You could keep track of the length of time it takes for each run of the programs to complete. Refining and optimising your code should yield higher run times.

For example:

```
class A {

            public static void main(String args[]) {

                   long start = System.currentTimeMillis();

                   //program does its thing

                   long finish = System.currentTimeMillis();

                   System.out.println("Time taken in ms " + Long.toString(finish - start));

            }

}
```

If you wish to use an array to store the grid for problem three feel free to copy the code from the next page and paste into Netbeans:

```
static int[][] matrix = {
        {8, 02, 22, 97, 38, 15, 00, 40, 00, 75, 04, 05, 07, 78, 52, 12, 50, 77, 91, 8},
        {49, 49, 99, 40, 17, 81, 18, 57, 60, 87, 17, 40, 98, 43, 69, 48, 04, 56, 62, 0},
        {81, 49, 31, 73, 55, 79, 14, 29, 93, 71, 40, 67, 53, 88, 30, 03, 49, 13, 36, 65},
        {52, 70, 95, 23, 04, 60, 11, 42, 69, 24, 68, 56, 01, 32, 56, 71, 37, 02, 36, 91},
        {22, 31, 16, 71, 51, 67, 63, 89, 41, 92, 36, 54, 22, 40, 40, 28, 66, 33, 13, 80},
        {24, 47, 32, 60, 99, 03, 45, 02, 44, 75, 33, 53, 78, 36, 84, 20, 35, 17, 12, 50},
        {32, 98, 81, 28, 64, 23, 67, 10, 26, 38, 40, 67, 59, 54, 70, 66, 18, 38, 64, 70},
        {67, 26, 20, 68, 02, 62, 12, 20, 95, 63, 94, 39, 63, 8, 40, 91, 66, 49, 94, 21},
        {24, 55, 58, 05, 66, 73, 99, 26, 97, 17, 78, 78, 96, 83, 14, 88, 34, 89, 63, 72},
        {21, 36, 23, 9, 75, 00, 76, 44, 20, 45, 35, 14, 00, 61, 33, 97, 34, 31, 33, 95},
        {78, 17, 53, 28, 22, 75, 31, 67, 15, 94, 03, 80, 04, 62, 16, 14, 9, 53, 56, 92},
        {16, 39, 05, 42, 96, 35, 31, 47, 55, 58, 88, 24, 00, 17, 54, 24, 36, 29, 85, 57},
        {86, 56, 00, 48, 35, 71, 89, 07, 05, 44, 44, 37, 44, 60, 21, 58, 51, 54, 17, 58},
        {19, 80, 81, 68, 05, 94, 47, 69, 28, 73, 92, 13, 86, 52, 17, 77, 04, 89, 55, 40},
        {04, 52, 8, 83, 97, 35, 99, 16, 07, 97, 57, 32, 16, 26, 26, 79, 33, 27, 98, 66},
        {88, 36, 68, 87, 57, 62, 20, 72, 03, 46, 33, 67, 46, 55, 12, 32, 63, 93, 53, 69},
        {04, 42, 16, 73, 38, 25, 39, 11, 24, 94, 72, 18, 8, 46, 29, 32, 40, 62, 76, 36},
        {20, 69, 36, 41, 72, 30, 23, 88, 34, 62, 99, 69, 82, 67, 59, 85, 74, 04, 36, 16},
        {20, 73, 35, 29, 78, 31, 90, 01, 74, 31, 49, 71, 48, 86, 81, 16, 23, 57, 05, 54},
        {01, 70, 54, 71, 83, 51, 54, 69, 16, 92, 33, 48, 61, 43, 52, 01, 89, 19, 67, 48}};
```