First year project
Project 99: Who's Julia?

Group: 99a
Simon Lehmann Knudsen, simkn15
Sonni Hedelund Jensen, sonje15
Asbjørn Mansa Jensen, asjen15
DM501

---

25. maj 2016

Indhold

1 Introduction

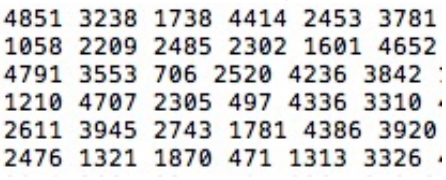
1.1 Julia

1.2 Features of Julia

1.3 Syntax differences

2 Projecteuler 11

In the 20x20 grid below, four numbers along a diagonal line have been marked in red. The product of these numbers is 26 x 63 x 78 x 14 = 1788696. What is the greatest product of four adjacent numbers in the same direction (up, down, left, right, or diagonally) in the 20x20 grid?



The algorithm first reads through a file with the input and makes a matrix. To run through a matrix, a nested for loop is needed, a total of two for loops. The outer loop runs through each row, and the inner loop runs through the columns. To calculate all the product in all directions there's 3 sets of for loops. 3 sets is consisting of an outer and an inner loop. The first set calculates the horizontal and vertical directions, second diagonally from left to right(downwards) and third set going diagonally from right to left(downwards). The algorithm starts in the most upper left cell, and going through all cells in the matrix. Lets have the first cell as (1, 1), first number is row(i) and second number is column(j). The first set which calculated horizontal and vertical would do the following on the first iteration:

```
1  for i = 1 : matLength
2    for j = 1 : matLength - numProd
3      #right/left
4      prod = 1
5      for k = 0 : numProd - 1
6      prod *= mat[i, j + k]
7      end
```

spacing

Horizontal: $(1,1) \cdot (1,2) \cdot (1,3) \cdot (1,4)$

Vertical: $(1,1) \cdot (2,1) \cdot (3,1) \cdot (4,1)$

More generally it will look like the following:

```
1  for i = 1 : matLength - numProd
2          for j = 1 : matLength - numProd
```

spacing

Horizontal: $(i, j+k) \cdot (i, j+k) \cdot (i, j+k) \cdot (i, j+k)$

Vertical: $(j+k, i) \cdot (j+k, i) \cdot (j+k, i) \cdot (j+k, i)$

In order to make the algorithm work on products with different amount of adjacent numbers(k), there's been added a for loop which loops **(k)** times, to match the number of adjacent numbers. With four adjacent numbers it will look like this, with starting at cell (1,1):

```
1  for i = 1 : matLength - numProd
2          for j = matLength : -1 : numProd
```

spacing

Horizontal: $(1, 1+0) \cdot (1, 1+1) \cdot (1, 1+2) \cdot (1, 1+2)$

Vertical: $(1+0, 1) \cdot (1+1, 1) \cdot (1+2, 1) \cdot (1+3, 1)$

After the first iteration the algorithm moves to next column, (i, j+1) until the last column is reached. Now going to next row, (i+1, j) and going through the columns once again until the last cell is reached. Last cell is the last cell which can still have **k** adjacent numbers. When $k = 4$ it stops 3 cells earlier than the last, since it can't go beyond the last column/row. The last horizontal product would be: $(end, end-3) \cdot (end, end-2) \cdot (end, end-1) \cdot (end, end)$. Diagonally from left to right looks like: spacing

$$(i+k, j+k) \cdot (i+k, j+k) \cdot (i+k, j+k) \cdot (i+k, j+k)$$

Diagonally from right to left starts at most upper right corner of matrix, and ending in the most lower left corner. The product of this looks like: spacing

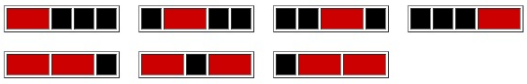
$$(i+k, j-k) \cdot (i+k, j-k) \cdot (i+k, j-k) \cdot (i+k, j-k)$$

For testing the algorithm on inputs larger than a $20 \cdot 20$ matrix, the numbers in each cell is 3-4 digits, the range of 100-9999. When making large matrices, e.g. a $5000 \cdot 5000$ many numbers would be the same if they were just 2 digits. Some languages make optimizations if it can predict what the result will be, e.g. multiplying many of the same numbers, and trying to avoid this by increasing the range of the numbers in each cell.

3 Projecteuler 116

A row of five black square tiles is to have a number of its tiles replaced with coloured oblong tiles chosen from red (length two), green (length three), or blue (length four).

If red tiles are chosen there are exactly seven ways this can be done.



If green tiles are chosen there are three ways.

And if blue tiles are chosen there are two ways.

Assuming that colours cannot be mixed there are 7 + 3 + 2 = 12 ways of replacing the black tiles in a row measuring five units in length.

How many different ways can the black tiles in a row measuring fifty units in length be replaced if colours cannot be mixed and at least one coloured tile must be used?

4 Quicksort

5 Statistics

6 Learning / Personal experience

7 Conclusion

8 Appendix (source code)