

Project Euler, ctd.

Dr. Mattox Beckman

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN
DEPARTMENT OF COMPUTER SCIENCE

Euler Problem 3 – Prime Factors

The prime factors of 13195 are 5, 7, 13, and 29. What is the largest prime factor of the number 600851475143?

Sectioning

```
1 *Main> plus a b = a + b
2 *Main> :t plus
3 plus :: Num a => a -> a -> a
4 *Main> plus 10 20
5 30
6 *Main> :t (plus 1)
7 (plus 1) :: Num a => a -> a
8 *Main> addTwo = plus 2
9 *Main> addTwo 10
10 12
```

- ▶ You can also say things like `(+1)` to get a partially applied operator.

The Sieve

- We will make something like the Sieve of Eratosthenes.

```
1 *Main> notDivides a n = n `mod` a /= 0
2 *Main> notDivides 2 10
3 False
4 *Main> notDivides 3 10
5 True
6 *Main> filter (notDivides 3) [1..10]
7 [1,2,4,5,7,8,10]
```

Go ahead and add the definition of `notDivides` to your file.

Building Up Lists

- ▶ The operator `:` creates a list from an element and another list.
- ▶ HASKELL `"a:b"` is like JAVA/C++ `"new Node(a,b)."`
- ▶ The built-in function `head` will get you the first element of a list.

```
1 *Main> 2 : filter (notDivides 2) [2..20]
2 [2,3,5,7,9,11,13,15,17,19]
3 *Main> 2 : filter (notDivides 2)
4           (3 : filter (notDivides 3) [2..20])
5 [2,3,5,7,11,13,17,19]
6 *Main> 2 : filter (notDivides 2)
7           (3 : filter (notDivides 3)
8             (5 : filter (notDivides 5) [2..20]))
9 [2,3,5,7,11,13,17,19]
```

- ▶ We need a recursive solution for this!

Making the Sieve

```
1 sieve (x:xs) = x : (sieve (filter (notDivides xs) xs))  
2  
3 primes = sieve [2..]
```

Sample Run

```
1 *Main> sieve [2..20]
2 [2,3,5,7,11,13,17,19,*** Exception: Prelude.head: empty list
3 *Main> take 20 (primes)
4 [2,3,5,7,11,13,17,19,23,29,31,37,41,43,47,53,59,61,67,71]
5 *Main> take 20 $ primes
6 [2,3,5,7,11,13,17,19,23,29,31,37,41,43,47,53,59,61,67,71]
```

Factors

- Now to get the factors ...

```
1 factors n = aux n primes
2   where aux 1 _      = []
3           aux n (p:ps) = case divMod n p of
4                           (n', 0) -> p : aux n' ps
5                           (_ , _) -> aux n ps
6
7 maxFactor n = foldr1 max $ factors n
8
9 euler3 = maxFactor 600851475143

1 *Main> foldr1 (+) [2,3,4,5]
2 14
3 *Main> euler3
4 6857
```


Problem 20 – Factorial Digit Sum

$n!$ means $n \times (n-1) \times \dots \times 3 \times 2 \times 1$.

For example, $10! = 10 \times 9 \times \dots \times 3 \times 2 \times 1 = 3628800$, and the sum of the digits in the number $10!$ is $3 + 6 + 2 + 8 + 8 + 0 + 0 = 27$.

Find the sum of the digits in the number $100!$

BigInts

- ▶ Most functional languages have “Big Integers,” constrained only by your computer’s memory.
- ▶ To get started, here’s the definition for factorial:

```
1 fact 0 = 1
2 func n = n * fact (n-1)
```

- ▶ If we run this on 100 it actually works!

```
1 *Main> fact 100  
2 933262154439441526816992388562667004907159682643816214685  
3 929638952175999932299156089414639761565182862536979208272  
4 23758251185210916864000000000000000000000000000
```

Divide and Conquer!

- ▶ To get the least significant digit, just take a modulus!
- ▶ To divide by 10 without remainder, just use `div`.

```
1 sumDigits 0 = 0
2 sumDigits n = n `mod` 10 + sumDigits (n `div` 10)
3
4 euler20 = sumDigits $ fact 100
```

Now try ...

```
1 *Main> euler20
2 648
```