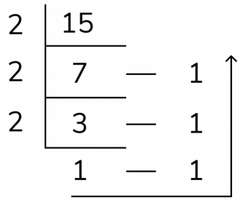
# Algorithms and Data Structures for DS

## Encoding

## Number encoding

Input: 15



Binary number: 1111

**Learning goals**

- Codes for numbers

## NUMBERS ON DIGITAL MACHINES

- Why do we care?
- Pretty much whatever we do in DS is implemented as some form of number crunching on a machine
- Computers perform billions of arithmetic operations per second
- But: Not all results are exact –
  even simple arithmetic may involve rounding or approximation
- Understanding how numbers are represented and processed is crucial for correct interpretation of results

## NUMBERS ON DIGITAL MACHINES

- The basic arithmetic operations are performed directly by the CPU
- The fewer bits per number, the faster
- Performance tradeoff: more precision often means slower computation
- To be efficient, numbers are encoded with a **fixed number of bytes**, so with N bits
- So we map infinite sets $\mathbb{Z}$ and $\mathbb{R}$ to $2^N$ machine numbers
- Such a mapping introduces rounding, overflow, and underflow
- Leads to potential loss of information.

## “BUG”-REPORT IN R I

```
To: R-bugs@biostat.ku.dk
Subject: error in trunc function

the command get a wrong result

> trunc(2.3 * 100)
[1] 229
```

Answer Duncan Murdoch:

```
That is the correct answer. 2.3 is not
representable exactly; the actual value used
is slightly less.
```

## “BUG”-REPORT IN R II

```
To: R-bugs@biostat.ku.dk
Subject: [Rd] match() (PR#13135)

The match function does not return value properly.
See an example below.

> a = seq(0.6, 1, by = 0.01)
> match(0.88, a)
[1] 29
> match(0.89, a)
[1] NA
```

Answer Brian Ripley:

```
FAQ Q7.31 strikes again!

0.89 is not a member of seq(0.6,1,by=0.01), since 0.01
cannot be represented exactly in a binary computer.
```

## “BUG”-REPORT IN R III

```
Subject: Re: Bug in R?
> Hi, I’m not sure if it’s really a bug:
> When you execute:
>> (2 / 3) * (0.6 / (1 - 0.6))
> the result will be:
> [1] 1
> but if you execute:
>> (2 / 3) * (0.6 / (1 - 0.6)) == 1
> the result is:
> [1] FALSE
> Note: I’m using version 2.9.2, (and tried it in 2.9.1 in 2.9.1 too)
with Microsoft Windows XP [Version 5.1.2600].
```

```
FAQ 7.31 strikes again:
R> 1 - (2 / 3) * (0.6 / (1 - 0.6))
[1] 2.220446e-16
R> .Machine$double.eps
[1] 2.220446e-16
```

## “BUG”-REPORT IN R IV

```
> Dear all,
>
> might seem an easy question but I cannot figure it out.
>
> floor(100 * (.58))
> [1] 57
>
> where is the trick here?
>   And how can I end up with the right answer?

See \texttt{R} FAQ 7.31
> sprintf("%.20f", 100 * .58)
[1] "57.99999999999999289457"
```