The intcalc package

Heiko Oberdiek*

2019/12/15 v1.3

Abstract

This package provides expandable arithmetic operations with integers.

Contents

*Please report any issues at https://github.com/ho-tex/intcalc/issues

1 Documentation

1.1 Introduction

Package intcalc defines arithmetic operations that deal with integers. Integers mean numbers in TEX. The same restrictions apply, the range is limited to [-2147483647, 2147483647].

The operations have the form of macros that take one or two integers as parameter and return the integer result. The macro name is a three letter operation name prefixed by the package name, e.g. `\intcalcAdd{10}{43}` returns 53.

The macros are fully expandable, exactly two expansion steps generate the result. Therefore the operations may be used nearly everywhere in TEX, even inside `\number`, `\csname`, file names, or other expandable contexts.

The package contains two implementations of the operations. If ε-TEX is detected then the macros are implemented using its features (`\numexpr`). Otherwise the slower implementation without ε-TEX's help is choosen.

1.2 Conditions

1.2.1 Preconditions

- Arguments can be anything that TEX interprets as "number". Examples: plain numbers, count or length register, macros that expands to a number.

- The arguments are limited to the range -2147483647 until 2147483647. These numbers belong to the range. Note that some operations have additionals restrictions to the range.
- The argument may be expressions that `\numexpr` understands if ε-TEX is available.
- The resulting number must fit in the allowed range.

1.2.2 Postconditions

Additional properties of the macros apart from calculating a correct result (of course ☺):

- The macros are fully expandable. Thus they can be used inside `\edef`, `\csname`, after `\number`, for example.
- Furthermore exactly two expansion steps calculate the result.
- The number consists of one optional minus sign and one to ten digits. The first digit is larger than zero for numbers that consists of more than one digit.

  In short, the number format is exactly the same as `\number` generates. And the tokens (minus sign, digits) have catcode 12 (other).
- Call by value is simulated. First the arguments are converted to numbers. Then these numbers are used in the calculations.

  Remember that arguments may contain expensive macros or ε-TEX expressions. This strategy avoids multiple evaluations of such arguments.

1.3 Error handling

There are two kinds of errors if a precondition is violated: Some errors are detected by the macros, example: division by zero. In this cases an undefined control sequence is called and causes a TeX error message, example: `\IntCalcError:DivisionByZero`. The name of the control sequence contains the reason for the error. The TEX error may be ignored. Then the operation returns zero as result. Because the macros are supposed to work in expandible contexts. An traditional error message, however, is not expandable and would break these contexts.

If a number exceeds the range of -2147483647 until 2147483647, then TEX throws an error "Number too big" and recovers by using biggest allowed value. Example for the negative number -3000000000 is replaced by -2147483647.

1.4 Operations

Some definition equations below use the function Int that converts a real number to an integer. The number is truncated that means rounding to zero:

$$\mathrm{Int}(x) := \begin{cases} \lfloor x \rfloor & \text{if } x \geq 0 \\ \lceil x \rceil & \text{otherwise} \end{cases}$$

1.4.1 Num

`\intcalcNum {⟨x⟩}`

Macro `\intcalcNum` converts its argument to a normalized integer number without unnecessary leading zeros or signs. The result matches the regular expression:

```
0|-?[1-9][0-9]*
```

1.4.2 Inv, Abs, Sgn

`\intcalcInv {⟨x⟩}`

Macro `\intcalcInv` switches the sign.

$$\mathrm{Inv}(x) := -x$$

`\intcalcAbs {⟨x⟩}`

Macro `\intcalcAbs` returns the absolute value of integer $\langle x \rangle$.

$$\mathrm{Abs}(x) := |x|$$

`\intcalcSgn {⟨x⟩}`

Macro `\intcalcSgn` encodes the sign of $\langle x \rangle$ as number.

$$\mathrm{Sgn}(x) := \begin{cases} -1 & \text{if } x < 0 \\ 0 & \text{if } x = 0 \\ 1 & \text{if } x > 0 \end{cases}$$

These return values can easily be distinguished by `\ifcase`:

```
\ifcase\intcalcSgn{<x>}
  $x=0$
\or
  $x>0$
\else
  $x<0$
\fi
```

1.4.3 Min, Max, Cmp

`\intcalcMin {⟨x⟩} {⟨y⟩}`

Macro `\intcalcMin` returns the smaller of the two integers.

$$\mathrm{Min}(x, y) := \begin{cases} x & \text{if } x < y \\ y & \text{otherwise} \end{cases}$$

`\intcalcMax {⟨x⟩} {⟨y⟩}`

Macro `\intcalcMax` returns the larger of the two integers.

$$\mathrm{Max}(x,y) := \begin{cases} x & \text{if } x > y \\ y & \text{otherwise} \end{cases}$$

`\intcalcCmp {⟨x⟩} {⟨y⟩}`

Macro `\intcalcCmp` encodes the comparison result as number:

$$\mathrm{Cmp}(x,y) := \begin{cases} -1 & \text{if } x < y \\ 0 & \text{if } x = y \\ 1 & \text{if } x > y \end{cases}$$

These values can be distinguished by `\ifcase`:

```
\ifcase\intcalcCmp{<x>}{<y>}
  $x=y$
\or
  $x>y$
\else
  $x<y$
\fi
```

1.4.4 Inc, Dec, Add, Sub

`\intcalcInc {⟨x⟩}`

Macro `\intcalcInc` increments $\langle x \rangle$ by one.

$$\mathrm{Inc}(x) := x + 1$$

`\intcalcDec {⟨x⟩}`

Macro `\intcalcDec` decrements $\langle x \rangle$ by one.

$$\mathrm{Dec}(x) := x - 1$$

`\intcalcAdd {⟨x⟩} {⟨y⟩}`

Macro `\intcalcAdd` adds the two numbers.

$$\mathrm{Add}(x,y) := x + y$$

`\intcalcSub {⟨x⟩} {⟨y⟩}`

Macro `\intcalcSub` calculates the difference.

$$\mathrm{Sub}(x,y) := x - y$$

1.4.5 Shl, Shr

`\intcalcShl {⟨x⟩}`

Macro `\intcalcShl` implements shifting to the left that means the number is multiplied by two. Overflow is possible. The sign is preserved.

$$\mathrm{Shl}(x) := x * 2$$

`\intcalcShr {⟨x⟩}`

Macro `\intcalcShr` implements shifting to the right. That is equivalent to an integer division by two. The sign is preserved.

$$\mathrm{Shr}(x) := \mathrm{Int}(x/2)$$

1.4.6 Mul, Sqr, Fac, Pow

`\intcalcMul {⟨x⟩} {⟨y⟩}`

Macro `\intcalcMul` calculates the product of $\langle x \rangle$ and $\langle y \rangle$.

$$\mathrm{Mul}(x, y) := x * y$$

`\intcalcSqr {⟨x⟩}`

Macro `\intcalcSqr` returns the square product.

$$\mathrm{Sqr}(x) := x^2$$

`\intcalcFac {⟨x⟩}`

Macro `\intcalcFac` returns the factorial of $\langle x \rangle$. Negative numbers are not permitted.

$$\mathrm{Fac}(x) := x! \qquad \text{for } x \geq 0$$

$(0! = 1)$

`\intcalcPow {⟨x⟩} {⟨y⟩}`

Macro `\intcalcPow` calculates the value of $\langle x \rangle$ to the power of $\langle y \rangle$. The error "division by zero" is thrown if $\langle x \rangle$ is zero and $\langle y \rangle$ is negative. permitted:

$$\mathrm{Pow}(x, y) := \mathrm{Int}(x^y) \qquad \text{for } x \neq 0 \text{ or } y \geq 0$$

$(0^0 = 1)$

1.4.7 Div, Mod

`\intcalcDiv` {⟨*x*⟩} {⟨*y*⟩}

Macro `\intcalcDiv` performs an integer division. Argument ⟨*y*⟩ must not be zero.

$$\mathrm{Div}(x, y) := \mathrm{Int}(x/y) \qquad \text{for } y \neq 0$$

`\intcalcMod` {⟨*x*⟩} {⟨*y*⟩}

Macro `\intcalcMod` gets the remainder of the integer division. The sign follows the divisor ⟨*y*⟩. Argument ⟨*y*⟩ must not be zero.

$$\mathrm{Mod}(x, y) := x \,\%\, y \qquad \text{for } y \neq 0$$

The result ranges:

$$-|y| < \mathrm{Mod}(x, y) \leq 0 \qquad \text{for } y < 0$$
$$0 \leq \mathrm{Mod}(x, y) < y \qquad \text{for } y \geq 0$$

1.5 Interface for programmer

If the programmer can ensure some more properties about the arguments of the operations, then the following macros are a little more efficient.

In general numbers must obey the following constraints:

- Plain number: digit tokens only, no command tokens.
- Non-negative. Signs are forbidden.
- Arguments and the result must fit in range `0..2147483647`.
- Delimited by exclamation mark. Curly braces around the number are not allowed and will break the code.

`\IntCalcInc` ⟨*number*⟩ `!`

Incrementation, range: `0..2147483646`.

`\IntCalcDec` ⟨*number*⟩ `!`

Decrementation, range: `1..2147483647`.

`\IntCalcAdd` ⟨*number A*⟩ `!` ⟨*number B*⟩ `!`

Addition, $A \geq B$.

`\IntCalcSub` ⟨*number A*⟩ `!` ⟨*number B*⟩ `!`

Subtraction, $A \geq B$.

`\IntCalcShl` ⟨*number*⟩ `!`

Left shift (multiplication with two), range: `0..1073741823`.

`\IntCalcShr` ⟨*number*⟩ `!`

Right shift (integer division by two).

`\IntCalcMul` ⟨*number A*⟩ `!` ⟨*number B*⟩ `!`

Multiplication, $A \geq B$.

`\IntCalcDiv` ⟨*number A*⟩ `!` ⟨*number B*⟩ `!`

Division operation.

`\IntCalcMod` ⟨*number A*⟩ `!` ⟨*number B*⟩ `!`

Modulo operation.

2 Implementation

1 ⟨*package⟩

2.1 Reload check and package identification

Reload check, especially if the package is not used with LaTeX.

```
2 \begingroup\catcode61\catcode48\catcode32=10\relax%
3   \catcode13=5 % ^^M
4   \endlinechar=13 %
5   \catcode35=6 % #
6   \catcode39=12 % '
7   \catcode44=12 % ,
8   \catcode45=12 % -
9   \catcode46=12 % .
10  \catcode58=12 % :
11  \catcode64=11 % @
12  \catcode123=1 % {
13  \catcode125=2 % }
14  \expandafter\let\expandafter\x\csname ver@intcalc.sty\endcsname
15  \ifx\x\relax % plain-TeX, first loading
16  \else
17    \def\empty{}%
18    \ifx\x\empty % LaTeX, first loading,
19      % variable is initialized, but \ProvidesPackage not yet seen
20    \else
21      \expandafter\ifx\csname PackageInfo\endcsname\relax
22        \def\x#1#2{%
23          \immediate\write-1{Package #1 Info: #2.}%
24        }%
25      \else
26        \def\x#1#2{\PackageInfo{#1}{#2, stopped}}%
27      \fi
28      \x{intcalc}{The package is already loaded}%
29      \aftergroup\endinput
30    \fi
31  \fi
32 \endgroup%
```

Package identification:

```
\begingroup\catcode61\catcode48\catcode32=10\relax%
  \catcode13=5 % ^^M
  \endlinechar=13 %
  \catcode35=6 % #
  \catcode39=12 % '
  \catcode40=12 % (
  \catcode41=12 % )
  \catcode44=12 % ,
  \catcode45=12 % -
  \catcode46=12 % .
  \catcode47=12 % /
  \catcode58=12 % :
  \catcode64=11 % @
  \catcode91=12 % [
  \catcode93=12 % ]
  \catcode123=1 % {
  \catcode125=2 % }
  \expandafter\ifx\csname ProvidesPackage\endcsname\relax
    \def\x#1#2#3[#4]{\endgroup
      \immediate\write-1{Package: #3 #4}%
      \xdef#1{#4}%
    }%
  \else
    \def\x#1#2[#3]{\endgroup
      #2[{#3}]%
      \ifx#1\@undefined
        \xdef#1{#3}%
      \fi
      \ifx#1\relax
        \xdef#1{#3}%
      \fi
    }%
  \fi
\expandafter\x\csname ver@intcalc.sty\endcsname
\ProvidesPackage{intcalc}%
  [2019/12/15 v1.3 Expandable calculations with integers (HO)]%
```

2.2 Catcodes

```
\begingroup\catcode61\catcode48\catcode32=10\relax%
  \catcode13=5 % ^^M
  \endlinechar=13 %
  \catcode123=1 % {
  \catcode125=2 % }
  \catcode64=11 % @
  \def\x{\endgroup
    \expandafter\edef\csname InCa@AtEnd\endcsname{%
      \endlinechar=\the\endlinechar\relax
      \catcode13=\the\catcode13\relax
      \catcode32=\the\catcode32\relax
      \catcode35=\the\catcode35\relax
      \catcode61=\the\catcode61\relax
      \catcode64=\the\catcode64\relax
      \catcode123=\the\catcode123\relax
      \catcode125=\the\catcode125\relax
    }%
  }%
```

```
\x\catcode61\catcode48\catcode32=10\relax%
\catcode13=5 % ^^M
\endlinechar=13 %
\catcode35=6 % #
\catcode64=11 % @
\catcode123=1 % {
\catcode125=2 % }
\def\TMP@EnsureCode#1#2{%
  \edef\InCa@AtEnd{%
    \InCa@AtEnd
    \catcode#1=\the\catcode#1\relax
  }%
  \catcode#1=#2\relax
}
\TMP@EnsureCode{33}{12}% !
\TMP@EnsureCode{40}{12}% (
\TMP@EnsureCode{41}{12}% )
\TMP@EnsureCode{42}{12}% *
\TMP@EnsureCode{43}{12}% +
\TMP@EnsureCode{45}{12}% -
\TMP@EnsureCode{47}{12}% /
\TMP@EnsureCode{58}{11}% : (letter!)
\TMP@EnsureCode{60}{12}% <
\TMP@EnsureCode{62}{12}% >
\TMP@EnsureCode{63}{14}% ? (comment!)
\edef\InCa@AtEnd{\InCa@AtEnd\noexpand\endinput}
\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname InCa@TestMode\endcsname\relax
\else
  \catcode63=9 % ? (ignore)
\fi
? \let\InCa@@TestMode\InCa@TestMode
```

2.3 Macros independent of ε-TEX

2.3.1 Abs, Sgn

\InCa@Abs

```
\def\InCa@Abs#1#2!{%
  \ifx#1-%
    #2%
  \else
    #1#2%
  \fi
}
```

\InCa@Sgn

```
\def\InCa@Sgn#1#2!{%
  \ifx#1-%
    -1%
  \else
    \ifx#10%
      0%
    \else
      1%
    \fi
  \fi
}
```

2.3.2 Min, Max, Cmp

\InCa@Min

```
\def\InCa@Min#1!#2!{%
  \ifnum#1<#2 %
    #1%
  \else
    #2%
  \fi
}
```

\InCa@Max

```
\def\InCa@Max#1!#2!{%
  \ifnum#1>#2 %
    #1%
  \else
    #2%
  \fi
}
```

\InCa@Cmp

```
\def\InCa@Cmp#1!#2!{%
  \ifnum#1=#2 %
    0%
  \else
    \ifnum#1<#2 %
      -%
    \fi
    1%
  \fi
}
```

2.3.3 Fac

\InCa@Fac It does not make much sense to calculate the faculty by an general algorithm. The allowed range of arguments is too low because of the limited integer domain.

```
\def\InCa@Fac#1!{%
  \ifcase#1 1% 0!
  \or 1% 1!
  \or 2% 2!
  \or 6% 3!
  \or 24% 4!
  \or 120% 5!
  \or 720% 6!
  \or 5040% 7!
  \or 40320% 8!
  \or 362880% 9!
  \or 3628800% 10!
  \or 39916800% 11!
  \or 479001600% 12!
  \else
    \ifnum#1<\z@
      0\IntCalcError:FacNegative%
    \else
      0\IntCalcError:FacOverflow%
    \fi
  \fi
}
```

2.4 Implementation based on ε-TEX

Only `\numexpr` is used from ε-TEX.

```
\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname numexpr\endcsname\relax
\else
```

2.4.1 Num

`\intcalcNum`

```
  \def\intcalcNum#1{%
    \the\numexpr#1\relax
  }%
```

2.4.2 Inv, Abs, Sgn

`\intcalcInv`

```
  \def\intcalcInv#1{%
    \number-\intcalcNum{#1} %
  }%
```

`\intcalcAbs`

```
  \def\intcalcAbs#1{%
    \number\expandafter\InCa@Abs\the\numexpr#1! %
  }%
```

`\intcalcSgn`

```
  \def\intcalcSgn#1{%
    \number\expandafter\InCa@Sgn\the\numexpr#1! %
  }%
```

2.4.3 Min, Max, Cmp

`\intcalcMin`

```
  \def\intcalcMin#1#2{%
    \number\expandafter\InCa@Min
    \the\numexpr#1\expandafter!%
    \the\numexpr#2! %
  }%
```

`\intcalcMax`

```
  \def\intcalcMax#1#2{%
    \number\expandafter\InCa@Max
    \the\numexpr#1\expandafter!%
    \the\numexpr#2! %
  }%
```

`\intcalcCmp`

```
  \def\intcalcCmp#1#2{%
    \number\expandafter\InCa@Cmp
    \the\numexpr#1\expandafter!\the\numexpr#2! %
  }%
```

2.4.4 Inc, Dec

\intcalcInc

```
\def\intcalcInc#1{%
  \the\numexpr#1+1\relax
}%
```

\intcalcDec

```
\def\intcalcDec#1{%
  \the\numexpr#1-1\relax
}%
```

\IntCalcInc

```
\def\IntCalcInc#1!{%
  \the\numexpr#1+1\relax
}%
```

\IntCalcDec

```
\def\IntCalcDec#1!{%
  \the\numexpr#1-1\relax
}%
```

2.4.5 Add, Sub

\intcalcAdd

```
\def\intcalcAdd#1#2{%
  \the\numexpr#1+(#2)\relax
}%
```

\intcalcSub

```
\def\intcalcSub#1#2{%
  \the\numexpr#1-(#2)\relax
}%
```

\IntCalcAdd

```
\def\IntCalcAdd#1!#2!{%
  \the\numexpr#1+#2\relax
}%
```

\IntCalcSub

```
\def\IntCalcSub#1!#2!{%
  \the\numexpr#1-#2\relax
}%
```

2.4.6 Shl, Shr

\intcalcShl

```
\def\intcalcShl#1{%
  \the\numexpr(#1)*2\relax
}%
```

\intcalcShr

```
\def\intcalcShr#1{%
  \number\expandafter\InCa@Shr\the\numexpr#1! %
}%
```

\IntCalcShl

```
\def\IntCalcShl#1!{%
  \the\numexpr#1*2\relax
}%
```

\IntCalcShr

```
\def\IntCalcShr#1!{%
  \the\numexpr\ifodd#1 (#1-1)\else#1\fi/2\relax
}%
```

\InCa@Shr

```
\def\InCa@Shr#1#2!{%
  \ifx#1-%
    -\InCa@Shr#2!%
  \else
    \ifodd#1#2 %
      \the\numexpr(#1#2-1)/2\relax
    \else
      \the\numexpr#1#2/2\relax
    \fi
  \fi
}%
```

2.4.7 Mul, Sqr, Fac

\intcalcMul

```
\def\intcalcMul#1#2{%
  \the\numexpr(#1)*(#2)\relax
}%
```

\IntCalcMul

```
\def\IntCalcMul#1!#2!{%
  \the\numexpr#1*#2\relax
}%
```

\intcalcSqr

```
\def\intcalcSqr#1{%
  \number\expandafter\InCa@Sqr\the\numexpr#1! %
}%
```

\InCa@Sqr

```
\def\InCa@Sqr#1!{%
  \the\numexpr#1*#1\relax
}%
```

\intcalcFac

```
\def\intcalcFac#1{%
  \number\expandafter\InCa@Fac\the\numexpr#1! %
}%
```

2.4.8 Pow

\intcalcPow

```
\def\intcalcPow#1#2{%
  \number\expandafter\InCa@Pow
  \the\numexpr#1\expandafter!%
  \the\numexpr#2! %
}%
```

\InCa@Pow

```
\def\InCa@Pow#1#2!#3#4!{%
  \ifcase#3#4 % power = 0
    1%
  \or % power = 1
    #1#2%
  \or % power = 2
    \the\numexpr#1#2*#1#2\relax
  \else
    \ifcase#1#2 % basis = 0, power <> 0
      0%
      \ifx#3-% power < 0
        0\IntCalcError:DivisionByZero%
      \fi
    \or
      1% basis = 1
    \else
      \ifnum#1#2=\m@ne % basis = -1
        \ifodd#3#4 %
          -%
        \fi
        1%
      \else % |basis| > 1
        \ifx#3-% power < 0
          0%
        \else % power > 2
          \InCa@PowRec#1#2!#3#4!1!%
        \fi
      \fi
    \fi
  \fi
}%
```

\InCa@PowRec

```
Pow(b, p) {
  PowRec(b, p, 1)
}
PowRec(b, p, r) {
  if p == 1 then
    return r*b
  else
    ifodd p then
      return PowRec(b*b, (p-1)/2, r*b) % p div 2 = (p-1)/2
    else
      return PowRec(b*b, (p-1)/2, r)
    fi
  fi
}
```

```
\def\InCa@PowRec#1!#2!#3!{%
  \ifnum#2=\@ne
    \the\numexpr#1*#3\relax
  \else
    \ifodd#2 %
      \expandafter\InCa@PowRec
      \the\numexpr#1*#1\expandafter!%
      \the\numexpr(#2-1)/2\expandafter!%
      \the\numexpr#1*#3\expandafter\expandafter\expandafter!%
    \else
      \expandafter\InCa@PowRec
```

```
      \the\numexpr#1*#1\expandafter!%
      \the\numexpr(#2-1)/2\expandafter!%
      \number#3\expandafter\expandafter\expandafter!%
    \fi
  \fi
}%
```

2.4.9 Div, Mod

TEX's `\divide` truncates, ε-TEX's `\numexpr` rounds the result of a division. The rounding method is called "Symmetric Arithmetic Rounding" or "Round-Half-Up" ("Kaufmännisches Runden" in German):

```
 1 =  3 divide 2 =  1.5 = numexpr  3/2 =  2
-1 = -3 divide 2 = -1.5 = numexpr -3/2 = -2
```

Macro `\intcalcDiv` follows TEX and truncates. The calculation is done by the following formula:

$$\mathrm{Div}(X,Y) = (X-(Y-1)/2)/Y \qquad \text{for } X,Y>0 \tag{1}$$

The operator '/' is `\numexpr`'s division.

`\intcalcDiv`

```
\def\intcalcDiv#1#2{%
  \number\expandafter\InCa@Div
  \the\numexpr#1\expandafter!%
  \the\numexpr#2! %
}%
```

`\InCa@Div`

```
\def\InCa@Div#1!#2!{%
  \ifcase#2 %
    0\IntCalcError:DivisionByZero%
  \else
    \ifcase#1 %
      0%
    \else
      \expandafter\InCa@@Div
      \romannumeral 0%
      \ifnum#1<\z@
        \expandafter-\number-#1%
      \else
        \expandafter+\number#1%
      \fi
      \expandafter!%
      \romannumeral 0%
      \ifnum#2<\z@
        \expandafter-\number-#2%
      \else
        \expandafter+\number#2%
      \fi
      !%
    \fi
  \fi
}%
```

`\IntCalcDiv`

```
\def\InCa@Temp#1{%
  \def\IntCalcDiv##1!##2!{%
    \number
    \ifcase##2 %
      0\IntCalcError:DivisionByZero%
    \else
      \ifcase##1 %
        0%
      \else
        \the\numexpr(##1-(##2-1)/2)/##2\relax
      \fi
    \fi
    #1%
  }%
}%
\InCa@Temp{ }%
```

\InCa@@Div

```
\def\InCa@@Div#1#2!#3#4!{%
  #1#3%
  \the\numexpr(#2-(#4-1)/2)/#4\relax
}%
```

\intcalcMod

```
\def\intcalcMod#1#2{%
  \number\expandafter\InCa@Mod
  \the\numexpr#1\expandafter!%
  \the\numexpr#2! %
}%
```

\InCa@Mod

```
\def\InCa@Mod#1!#2!{%
  \ifcase#2 %
    0\IntCalcError:DivisionByZero%
  \else
    \ifcase#1 %
      0%
    \else
      \expandafter\InCa@@Mod
      \romannumeral 0%
      \ifnum#1<\z@
        \expandafter-\number-#1%
      \else
        \expandafter+\number#1%
      \fi
      \expandafter!%
      \romannumeral 0%
      \ifnum#2<\z@
        \expandafter-\number-#2%
      \else
        \expandafter+\number#2%
      \fi
      !%
    \fi
  \fi
}%
```

\IntCalcMod

```
  \def\InCa@Temp#1{%
    \def\IntCalcMod##1!##2!{%
      \number
      \ifcase##2 %
        0\IntCalcError:DivisionByZero%
      \else
        \ifcase##1 %
          0%
        \else
          \the\numexpr##1-(##1-(##2-1)/2)/##2*##2\relax
        \fi
      \fi
      #1%
    }%
  }%
  \InCa@Temp{ }%
```

`\InCa@@Mod`

```
  \def\InCa@@Mod#1#2!#3#4!{%
    \if#3+%
      \if#1+%
        \the\numexpr#2-\InCa@@Div+#2!+#4!*#4\relax
      \else
        \expandafter\InCa@ModX
        \the\numexpr-#2+\InCa@@Div+#2!+#4!*#4!#4!%
      \fi
    \else
      -%
      \if#1+%
        \expandafter\InCa@ModX
        \the\numexpr-#2+\InCa@@Div+#2!+#4!*#4!#4!%
      \else
        \the\numexpr#2-\InCa@@Div+#2!+#4!*#4\relax
      \fi
    \fi
  }%
```

`\InCa@ModX`

```
  \def\InCa@ModX#1!#2!{%
    \ifcase#1 %
      0%
    \else
      \the\numexpr#1+#2\relax
    \fi
  }%

  \expandafter\InCa@AtEnd
\fi%
```

2.5 Implementation without ε-TEX

2.5.1 Num

`\intcalcNum`

```
\def\intcalcNum#1{%
  \number\expandafter\InCa@FirstOfOne\number#1! %
}
```

2.5.2 Inv, Abs, Sgn

\intcalcInv

```
\def\intcalcInv#1{%
  \number\expandafter\InCa@FirstOfOne\number-#1! %
}
```

\InCa@FirstOfOne

```
\def\InCa@FirstOfOne#1!{#1}
```

\intcalcAbs

```
\def\intcalcAbs#1{%
  \number\expandafter\InCa@Abs\number#1! %
}
```

\intcalcSgn

```
\def\intcalcSgn#1{%
  \number\expandafter\InCa@Sgn\number#1! %
}
```

2.5.3 Min, Max, Cmp

\intcalcMin

```
\def\intcalcMin#1#2{%
  \number\expandafter\InCa@Min
  \number\number#1\expandafter!\number#2! %
}
```

\intcalcMax

```
\def\intcalcMax#1#2{%
  \number\expandafter\InCa@Max
  \number\number#1\expandafter!\number#2! %
}
```

\intcalcCmp

```
\def\intcalcCmp#1#2{%
  \number\expandafter\InCa@Cmp
  \number\number#1\expandafter!\number#2! %
}%
```

2.5.4 Inc, Dec

\intcalcInc

```
\def\intcalcInc#1{%
  \number\expandafter\InCa@IncSwitch\number#1! %
}
```

\InCa@IncSwitch

```
\def\InCa@IncSwitch#1#2!{%
  \ifx#1-%
    -%
    \csname InCa@Empty%
    \InCa@Dec#2!%
  \else
    \csname InCa@Empty%
    \InCa@Inc#1#2!%
  \fi
}
```

\intcalcDec

```
\def\intcalcDec#1{%
  \number\expandafter\InCa@DecSwitch\number#1! %
}
```

\InCa@DecSwitch

```
\def\InCa@DecSwitch#1#2!{%
  \ifx#1-%
    -%
    \csname InCa@Empty%
    \expandafter\InCa@Inc#2!%
  \else
    \ifx#10%
      -1%
    \else
      \csname InCa@Empty%
      \InCa@Dec#1#2!%
    \fi
  \fi
}
```

\IntCalcInc

```
\def\IntCalcInc#1!{%
  \number\csname InCa@Empty\InCa@Inc#1! %
}
```

\IntCalcDec

```
\def\IntCalcDec#1!{%
  \number\csname InCa@Empty\InCa@Dec#1! %
}
```

\InCa@Inc

```
\def\InCa@Inc#1#2{%
  \ifx#2!%
    \csname InCa@IncDigit#1\endcsname1%
  \else
    \csname InCa@IncDigit#1%
    \expandafter\InCa@Inc\expandafter#2%
  \fi
}
```

\InCa@IncDigit[0-8]

```
\def\InCa@Temp#1#2{%
  \expandafter\def\csname InCa@IncDigit#1\endcsname##1{%
    \endcsname
    0%
    \ifcase##1 %
      #1%
    \else
      #2%
    \fi
  }%
}
\InCa@Temp 01
\InCa@Temp 12
\InCa@Temp 23
\InCa@Temp 34
```

```
\InCa@Temp 45
\InCa@Temp 56
\InCa@Temp 67
\InCa@Temp 78
\InCa@Temp 89
```

\InCa@IncDigit9

```
\expandafter\def\csname InCa@IncDigit9\endcsname#1{%
  \expandafter\endcsname
  \ifcase#1 %
    09%
  \else
    10%
  \fi
}
```

\InCa@Dec

```
\def\InCa@Dec#1#2{%
  \ifx#2!%
    \csname InCa@DecDigit#1\endcsname1%
  \else
    \csname InCa@DecDigit#1%
    \expandafter\InCa@Dec\expandafter#2%
  \fi
}
```

\InCa@DecDigit[1-9]

```
\def\InCa@Temp#1#2{%
  \expandafter\def\csname InCa@DecDigit#1\endcsname##1{%
    \endcsname
    0%
    \ifcase##1 %
      #1%
    \else
      #2%
    \fi
  }%
}
\InCa@Temp 98
\InCa@Temp 87
\InCa@Temp 76
\InCa@Temp 65
\InCa@Temp 54
\InCa@Temp 43
\InCa@Temp 32
\InCa@Temp 21
\InCa@Temp 10
```

\InCa@DecDigit0

```
\expandafter\def\csname InCa@DecDigit0\endcsname#1{%
  \expandafter\endcsname
  \ifcase#1 %
    00%
  \else
    19%
  \fi
}
```

2.5.5 Add, Sub

\intcalcAdd

```
\def\intcalcAdd#1#2{%
  \number
    \expandafter\InCa@AddSwitch
    \number\number#1\expandafter!%
    \number#2! %
}
```

\intcalcSub

```
\def\intcalcSub#1#2{%
  \number
    \expandafter\InCa@AddSwitch
    \number\number#1\expandafter!%
    \number-\number#2! %
}
```

\InCa@AddSwitch Decision table for `\InCa@AddSwitch`. The sign of negative numbers can be removed by a simple `\@gobble` instead of the more expensive `\number-`.

$x < 0$	$y < 0$	$x < y$	$-$	$\mathrm{Add}(-x, -y)$
		else		$\mathrm{Add}(-y, -x)$
	else	$-x > y$	$-$	$\mathrm{Sub}(-x, y)$
		else	$+$	$\mathrm{Sub}(y, -x)$
else	$y < 0$	$x > -y$	$+$	$\mathrm{Sub}(x, -y)$
		else	$-$	$\mathrm{Sub}(-y, x)$
	else	$x > y$	$+$	$\mathrm{Add}(x, y)$
		else		$\mathrm{Add}(y, x)$

```
\def\InCa@AddSwitch#1!#2!{%
  \ifnum#1<\z@
    \ifnum#2<\z@
      -%
      \ifnum#1<#2 %
        \expandafter\InCa@Add\number-#1\expandafter!%
        \@gobble#2!%
      \else
        \expandafter\InCa@Add\number-#2\expandafter!%
        \@gobble#1!%
      \fi
    \else
      \ifnum-#1>#2 %
        -%
        \expandafter\InCa@Sub\@gobble#1!#2!%
      \else
        \expandafter\InCa@Sub\number#2\expandafter!%
        \@gobble#1!%
      \fi
    \fi
  \else
    \ifnum#2<\z@
      \ifnum#1>-#2 %
        \expandafter\InCa@Sub\number#1\expandafter!%
        \@gobble#2!%
      \else
        -%
```

```
        \expandafter\InCa@Sub\@gobble#2!#1!%
      \fi
    \else
      \ifnum#1>#2 %
        \InCa@Add#1!#2!%
      \else
        \InCa@Add#2!#1!%
      \fi
    \fi
  \fi
}
```

\IntCalcAdd

```
\def\IntCalcAdd#1!#2!{%
  \number\InCa@Add#1!#2! %
}
```

\IntCalcSub

```
\def\IntCalcSub#1!#2!{%
  \number\InCa@Sub#1!#2! %
}
```

\InCa@Space

```
\begingroup
  \def\x#1{\endgroup
    \let\InCa@Space= #1%
  }%
\x{ }
```

\InCa@Add

```
\def\InCa@Add#1!#2!{%
  \ifcase#2 %
    #1%
  \else
    \InCa@@Add#1!#2!000000000\InCa@Space
  \fi
}
```

\InCa@Sub

```
\def\InCa@Sub#1!#2!{%
  \ifnum#1=#2 %
    0%
  \else
    \InCa@@Sub#1!#2!000000000\InCa@Space
  \fi
}
```

\InCa@@Add

```
\def\InCa@@Add#1!#2#3!{%
  \ifx\InCa@Empty#3\InCa@Empty
    \@ReturnAfterElseFi{%
      \InCa@@@Add!!#1!#2%
    }%
  \else
    \@ReturnAfterFi{%
      \InCa@@Add#1!#3!#2%
    }%
  \fi
}
```

\InCa@@Sub

```
\def\InCa@@Sub#1!#2#3!{%
  \ifx\InCa@Empty#3\InCa@Empty
    \@ReturnAfterElseFi{%
      \InCa@@@Sub!!#1!#2%
    }%
  \else
    \@ReturnAfterFi{%
      \InCa@@Sub#1!#3!#2%
    }%
  \fi
}
```

\InCa@@@Add

```
\def\InCa@@@Add#1!#2!#3#4!#5{%
  \ifx\InCa@Empty#4\InCa@Empty
    \csname InCa@Empty%
    \@ReturnAfterElseFi{%
      \InCa@ProcessAdd#1#3!#5#2%
    }%
  \else
    \@ReturnAfterFi{%
      \InCa@@@Add#1#3!#5#2!#4!%
    }%
  \fi
}
```

\InCa@@@Sub

```
\def\InCa@@@Sub#1!#2!#3#4!#5{%
  \ifx\InCa@Empty#4\InCa@Empty
    \csname @gobble%
    \@ReturnAfterElseFi{%
      \InCa@ProcessSub#1#3!#5#2%
    }%
  \else
    \@ReturnAfterFi{%
      \InCa@@@Sub#1#3!#5#2!#4!%
    }%
  \fi
}
```

\InCa@ProcessAdd

```
\def\InCa@ProcessAdd#1#2!#3#4{%
  \ifx\InCa@Empty#2\InCa@Empty
    \csname InCa@AddDigit#1\endcsname#3%
    \romannumeral0#4%
  \else
    \csname InCa@AddDigit#1\csname InCa@DigitCarry#3%
    \@ReturnAfterFi{%
      \InCa@ProcessAdd#2!#4%
    }%
  \fi
}
```

\InCa@ProcessSub

```
\def\InCa@ProcessSub#1#2!#3#4{%
  \ifx\InCa@Empty#2\InCa@Empty
    \csname InCa@SubDigit#1\endcsname#3%
```

```
    \romannumeral0#4%
  \else
    \csname InCa@SubDigit#1\csname InCa@DigitCarry#3%
    \@ReturnAfterFi{%
      \InCa@ProcessSub#2!#4%
    }%
  \fi
}
```

\InCa@DigitCarry[0-9]

```
\def\InCa@Temp#1#2{%
  \expandafter\def\csname InCa@DigitCarry#1\endcsname##1{%
    \ifcase##1 %
      \endcsname#1%
    \else
      \endcsname#2%
    \fi
  }%
}
\InCa@Temp 01
\InCa@Temp 12
\InCa@Temp 23
\InCa@Temp 34
\InCa@Temp 45
\InCa@Temp 56
\InCa@Temp 67
\InCa@Temp 78
\InCa@Temp 89
\InCa@Temp 9{{10}}
```

\InCa@AddDigit0

```
\expandafter\def\csname InCa@AddDigit0\endcsname#1{%
  \ifnum#1>9 %
    \endcsname10%
  \else
    \endcsname0#1%
  \fi
}
```

\InCa@AddDigit[1-9]

```
\def\InCa@Temp#1#2#3{%
  \expandafter\def\csname InCa@AddDigit#1\endcsname##1{%
    \ifnum##1>#2 %
      \endcsname 1%
    \else
      \endcsname 0%
    \fi
    \ifcase##1 #1% 0
    #3%
    \else #1% 10
    \fi
  }%
}
\InCa@Temp 18{%
  \or 2% 1
  \or 3% 2
  \or 4% 3
  \or 5% 4
```

```
  \or 6% 5
  \or 7% 6
  \or 8% 7
  \or 9% 8
  \or 0% 9
}%
\InCa@Temp 27{%
  \or 3% 1
  \or 4% 2
  \or 5% 3
  \or 6% 4
  \or 7% 5
  \or 8% 6
  \or 9% 7
  \or 0% 8
  \or 1% 9
}%
\InCa@Temp 36{%
  \or 4% 1
  \or 5% 2
  \or 6% 3
  \or 7% 4
  \or 8% 5
  \or 9% 6
  \or 0% 7
  \or 1% 8
  \or 2% 9
}%
\InCa@Temp 45{%
  \or 5% 1
  \or 6% 2
  \or 7% 3
  \or 8% 4
  \or 9% 5
  \or 0% 6
  \or 1% 7
  \or 2% 8
  \or 3% 9
}%
\InCa@Temp 54{%
  \or 6% 1
  \or 7% 2
  \or 8% 3
  \or 9% 4
  \or 0% 5
  \or 1% 6
  \or 2% 7
  \or 3% 8
  \or 4% 9
}%
\InCa@Temp 63{%
  \or 7% 1
  \or 8% 2
  \or 9% 3
  \or 0% 4
  \or 1% 5
  \or 2% 6
  \or 3% 7
```

```
  \or 4% 8
  \or 5% 9
}%
\InCa@Temp 72{%
  \or 8% 1
  \or 9% 2
  \or 0% 3
  \or 1% 4
  \or 2% 5
  \or 3% 6
  \or 4% 7
  \or 5% 8
  \or 6% 9
}%
\InCa@Temp 81{%
  \or 9% 1
  \or 0% 2
  \or 1% 3
  \or 2% 4
  \or 3% 5
  \or 4% 6
  \or 5% 7
  \or 6% 8
  \or 7% 9
}%
\InCa@Temp 90{%
  \or 0% 1
  \or 1% 2
  \or 2% 3
  \or 3% 4
  \or 4% 5
  \or 5% 6
  \or 6% 7
  \or 7% 8
  \or 8% 9
}%
```

`\InCa@SubDigit[0-9]`

```
\def\InCa@Temp#1#2{%
  \expandafter\def\csname InCa@SubDigit#1\endcsname##1{%
    \ifnum##1>#1 %
      \endcsname 1%
    \else
      \endcsname 0%
    \fi
    \ifcase##1 #1% 0
    #2%
    \else #1% 10
    \fi
  }%
}
\InCa@Temp 0{%
  \or 9% 1
  \or 8% 2
  \or 7% 3
  \or 6% 4
  \or 5% 5
  \or 4% 6
```

```
  \or 3% 7
  \or 2% 8
  \or 1% 9
}
\InCa@Temp 1{%
  \or 0% 1
  \or 9% 2
  \or 8% 3
  \or 7% 4
  \or 6% 5
  \or 5% 6
  \or 4% 7
  \or 3% 8
  \or 2% 9
}
\InCa@Temp 2{%
  \or 1% 1
  \or 0% 2
  \or 9% 3
  \or 8% 4
  \or 7% 5
  \or 6% 6
  \or 5% 7
  \or 4% 8
  \or 3% 9
}
\InCa@Temp 3{%
  \or 2% 1
  \or 1% 2
  \or 0% 3
  \or 9% 4
  \or 8% 5
  \or 7% 6
  \or 6% 7
  \or 5% 8
  \or 4% 9
}
\InCa@Temp 4{%
  \or 3% 1
  \or 2% 2
  \or 1% 3
  \or 0% 4
  \or 9% 5
  \or 8% 6
  \or 7% 7
  \or 6% 8
  \or 5% 9
}
\InCa@Temp 5{%
  \or 4% 1
  \or 3% 2
  \or 2% 3
  \or 1% 4
  \or 0% 5
  \or 9% 6
  \or 8% 7
  \or 7% 8
  \or 6% 9
```

```
}
\InCa@Temp 6{%
  \or 5% 1
  \or 4% 2
  \or 3% 3
  \or 2% 4
  \or 1% 5
  \or 0% 6
  \or 9% 7
  \or 8% 8
  \or 7% 9
}
\InCa@Temp 7{%
  \or 6% 1
  \or 5% 2
  \or 4% 3
  \or 3% 4
  \or 2% 5
  \or 1% 6
  \or 0% 7
  \or 9% 8
  \or 8% 9
}
\InCa@Temp 8{%
  \or 7% 1
  \or 6% 2
  \or 5% 3
  \or 4% 4
  \or 3% 5
  \or 2% 6
  \or 1% 7
  \or 0% 8
  \or 9% 9
}
\InCa@Temp 9{%
  \or 8% 1
  \or 7% 2
  \or 6% 3
  \or 5% 4
  \or 4% 5
  \or 3% 6
  \or 2% 7
  \or 1% 8
  \or 0% 9
}
```

2.5.6 Shl, Shr

`\intcalcShl`

```
\def\intcalcShl#1{%
  \number\expandafter\InCa@ShlSwitch\number#1! %
}
```

`\InCa@ShlSwitch`

```
\def\InCa@ShlSwitch#1#2!{%
  \ifx#1-%
    -\csname InCa@Empty%
    \InCa@Shl#2!%
```

```
  \else
    \csname InCa@Empty%
    \InCa@Shl#1#2!%
  \fi
}
```

\IntCalcShl

```
\def\IntCalcShl#1!{%
  \number
  \csname InCa@Empty%
  \InCa@Shl#1! %
}
```

\IntCal@ShlDigit

```
\def\InCa@Shl#1#2{%
  \ifx#2!%
    \csname InCa@ShlDigit#1\endcsname0%
  \else
    \csname InCa@ShlDigit#1%
    \@ReturnAfterFi{%
      \InCa@Shl#2%
    }%
  \fi
}
```

\InCa@ShlDigit0

```
\expandafter\def\csname InCa@ShlDigit0\endcsname{%
  \endcsname0%
}
```

\InCa@ShlDigit[1-9]

```
\def\InCa@Temp#1#2#3#4#5{%
  \expandafter\def\csname InCa@ShlDigit#1\endcsname##1{%
    \expandafter\endcsname
    \ifcase##1 %
      #2#3%
    \else
      #4#5%
    \fi
  }%
}
\InCa@Temp 10203
\InCa@Temp 20405
\InCa@Temp 30607
\InCa@Temp 40809
\InCa@Temp 51011
\InCa@Temp 61213
\InCa@Temp 71415
\InCa@Temp 81617
\InCa@Temp 91819
```

\intcalcShr

```
\def\intcalcShr#1{%
  \number\expandafter\InCa@ShrSwitch\number#1! %
}
```

\InCa@ShrSwitch

```
\def\InCa@ShrSwitch#1#2!{%
  \ifx#1-%
    -\InCa@Shr#2!%
  \else
    \InCa@Shr#1#2!%
  \fi
}
```

\IntCalcShr

```
\def\IntCalcShr#1!{%
  \number\InCa@Shr#1! %
}
```

\InCa@Shr

```
\def\InCa@Shr#1#2{%
  \InCa@ShrDigit#1!%
  \ifx#2!%
  \else
    \@ReturnAfterFi{%
      \ifodd#1 %
        \@ReturnAfterElseFi{%
          \InCa@Shr{1#2}%
        }%
      \else
        \expandafter\InCa@Shr\expandafter#2%
      \fi
    }%
  \fi
}
```

```
\def\InCa@ShrDigit#1!{%
  \ifcase#1 0% 0
  \or 0% 1
  \or 1% 2
  \or 1% 3
  \or 2% 4
  \or 2% 5
  \or 3% 6
  \or 3% 7
  \or 4% 8
  \or 4% 9
  \or 5% 10
  \or 5% 11
  \or 6% 12
  \or 6% 13
  \or 7% 14
  \or 7% 15
  \or 8% 16
  \or 8% 17
  \or 9% 18
  \or 9% 19
  \fi
}
```

2.5.7 \InCa@Tim

\InCa@Tim Macro \InCa@Tim implements "Number *tim*es digit".

```
\def\InCa@Temp#1{%
```

```
  \def\InCa@Tim##1##2{%
    \number
      \ifcase##2 % 0
        0%
      \or % 1
        ##1%
      \else % 2-9
        \csname InCa@Empty%
        \InCa@ProcessTim##2##1!%
      \fi
    #1%
  }%
}
\InCa@Temp{ }
```

\InCa@ProcessTim

```
\def\InCa@ProcessTim#1#2#3{%
  \ifx#3!%
    \csname InCa@TimDigit#2\endcsname#10%
  \else
    \csname InCa@TimDigit#2\csname InCa@Param#1%
    \@ReturnAfterFi{%
      \InCa@ProcessTim#1#3%
    }%
  \fi
}
```

\InCa@Param[0-9]

```
\def\InCa@Temp#1{%
  \expandafter\def\csname InCa@Param#1\endcsname{%
    \endcsname#1%
  }%
}
\InCa@Temp 0%
\InCa@Temp 1%
\InCa@Temp 2%
\InCa@Temp 3%
\InCa@Temp 4%
\InCa@Temp 5%
\InCa@Temp 6%
\InCa@Temp 7%
\InCa@Temp 8%
\InCa@Temp 9%
```

\InCa@TimDigit0

```
\expandafter\def\csname InCa@TimDigit0\endcsname#1#2{%
  \endcsname
  0#2%
}
```

\InCa@TimDigit1

```
\expandafter\def\csname InCa@TimDigit1\endcsname#1#2{%
  \ifcase#2 %
    \endcsname 0#1%
  \else
    \csname InCa@AddDigit#1\endcsname #2%
  \fi
}
```

\InCa@TimDigit[2-9]

```
\def\InCa@Temp#1#2{%
  \expandafter\def\csname InCa@TimDigit#1\endcsname##1{%
    \expandafter\InCa@TimDigitCarry
    \number
      \ifcase##1 0% 0
      #2%
      \fi
    !%
  }%
}
\InCa@Temp 2{%
  \or  2% 1
  \or  4% 2
  \or  6% 3
  \or  8% 4
  \or 10% 5
  \or 12% 6
  \or 14% 7
  \or 16% 8
  \or 18% 9
}
\InCa@Temp 3{%
  \or  3% 1
  \or  6% 2
  \or  9% 3
  \or 12% 4
  \or 15% 5
  \or 18% 6
  \or 21% 7
  \or 24% 8
  \or 27% 9
}
\InCa@Temp 4{%
  \or  4% 1
  \or  8% 2
  \or 12% 3
  \or 16% 4
  \or 20% 5
  \or 24% 6
  \or 28% 7
  \or 32% 8
  \or 36% 9
}
\InCa@Temp 5{%
  \or  5% 1
  \or 10% 2
  \or 15% 3
  \or 20% 4
  \or 25% 5
  \or 30% 6
  \or 35% 7
  \or 40% 8
  \or 45% 9
}
\InCa@Temp 6{%
  \or  6% 1
  \or 12% 2
```

```
  \or 18% 3
  \or 24% 4
  \or 30% 5
  \or 36% 6
  \or 42% 7
  \or 48% 8
  \or 54% 9
}
\InCa@Temp 7{%
  \or  7% 1
  \or 14% 2
  \or 21% 3
  \or 28% 4
  \or 35% 5
  \or 42% 6
  \or 49% 7
  \or 56% 8
  \or 63% 9
}
\InCa@Temp 8{%
  \or  8% 1
  \or 16% 2
  \or 24% 3
  \or 32% 4
  \or 40% 5
  \or 48% 6
  \or 56% 7
  \or 64% 8
  \or 72% 9
}
\InCa@Temp 9{%
  \or  9% 1
  \or 18% 2
  \or 27% 3
  \or 36% 4
  \or 45% 5
  \or 54% 6
  \or 63% 7
  \or 72% 8
  \or 81% 9
}
```

\InCa@TimDigitCarry

```
\def\InCa@TimDigitCarry#1!{%
  \ifnum#1<10 %
    \csname InCa@AddDigit#1\expandafter\endcsname
  \else
    \@ReturnAfterFi{%
      \InCa@@TimDigitCarry#1!%
    }%
  \fi
}
```

\InCa@@TimDigitCarry

```
\def\InCa@@TimDigitCarry#1#2!#3{%
  \csname InCa@DigitCarry#1%
  \csname InCa@AddDigit#2\endcsname #3%
}
```

2.5.8 Mul

\intcalcMul

```
\def\intcalcMul#1#2{%
  \number
    \expandafter\InCa@MulSwitch
    \number\number#1\expandafter!%
    \number#2! %
}
```

\InCa@MulSwitch Decision table for \InCa@MulSwitch.

$x < 0$	$y < 0$	$x < y$	$+$	$\mathrm{Mul}(-x, -y)$
		else		$\mathrm{Mul}(-y, -x)$
	else	$-x > y$	$-$	$\mathrm{Mul}(-x, y)$
		else		$\mathrm{Mul}(y, -x)$
else	$y < 0$	$x > -y$	$-$	$\mathrm{Mul}(x, -y)$
		else		$\mathrm{Mul}(-y, x)$
	else	$x > y$	$+$	$\mathrm{Mul}(x, y)$
		else		$\mathrm{Mul}(y, x)$

```
\def\InCa@MulSwitch#1!#2!{%
  \ifnum#1<\z@
    \ifnum#2<\z@
      \ifnum#1<#2 %
        \expandafter\InCa@Mul\number-#1\expandafter!%
        \@gobble#2!%
      \else
        \expandafter\InCa@Mul\number-#2\expandafter!%
        \@gobble#1!%
      \fi
    \else
      -%
      \ifnum-#1>#2 %
        \expandafter\InCa@Mul\@gobble#1!#2!%
      \else
        \expandafter\InCa@Mul\number#2\expandafter!%
        \@gobble#1!%
      \fi
    \fi
  \else
    \ifnum#2<\z@
      -%
      \ifnum#1>-#2 %
        \expandafter\InCa@Mul\number#1\expandafter!%
        \@gobble#2!%
      \else
        \expandafter\InCa@Mul\@gobble#2!#1!%
      \fi
    \else
      \ifnum#1>#2 %
        \InCa@Mul#1!#2!%
      \else
        \InCa@Mul#2!#1!%
      \fi
    \fi
  \fi
}
```

\IntCalcMul

```
\def\IntCalcMul#1!#2!{%
  \number\InCa@Mul#1!#2! %
}
```

\InCa@Mul

```
\def\InCa@Mul#1!#2!{%
  \ifcase#2 %
    0%
  \or
    #1%
  \or
    \csname InCa@Empty%
    \expandafter\InCa@Shl#1!%
  \else
    \ifnum#2<10 %
      \InCa@Tim{#1}#2%
    \else
      \InCa@ProcessMul!#2!#1!%
    \fi
  \fi
}
```

\InCa@Mul

```
\def\InCa@ProcessMul#1!#2#3!#4!{%
  \ifx\InCa@Empty#3\InCa@Empty
    \expandafter\InCa@Add\number
    #10\expandafter\expandafter\expandafter!%
    \InCa@Tim{#4}#2!%
  \else
    \ifx\InCa@Empty#1\InCa@Empty
      \expandafter\expandafter\expandafter\InCa@ProcessMul
      \InCa@Tim{#4}#2!%
      #3!#4!%
    \else
      \expandafter\InCa@ProcessMul\number
      \expandafter\InCa@Add\number%
      #10\expandafter\expandafter\expandafter!%
      \InCa@Tim{#4}#2!!%
      #3!#4!%
    \fi
  \fi
}
```

2.5.9 Sqr, Fac

\intcalcSqr

```
\def\intcalcSqr#1{%
  \number\expandafter\InCa@Sqr\number#1! %
}
```

\InCa@Sqr

```
\def\InCa@Sqr#1#2!{%
  \ifx#1-%
    \InCa@Mul#2!#2!%
  \else
    \InCa@Mul#1#2!#1#2!%
```

```
  \fi
}
```

\intcalcFac

```
\def\intcalcFac#1{%
  \number\expandafter\InCa@Fac\number#1! %
}
```

2.5.10 Pow

\intcalcPow

```
\def\intcalcPow#1#2{%
  \number\expandafter\InCa@Pow
  \number\number#1\expandafter!%
  \number#2! %
}
```

\InCa@Pow

```
\def\InCa@Pow#1#2!#3#4!{%
  \ifcase#3#4 % power = 0
    1%
  \or % power = 1
    #1#2%
  \or % power = 2
    \ifx#1-%
      \InCa@Mul#2!#2!%
    \else
      \InCa@Mul#1#2!#1#2!%
    \fi
  \else
    \ifcase#1#2 % basis = 0, power <> 0
      0%
      \ifx#3-% power < 0
        0\IntCalcError:DivisionByZero%
      \fi
    \or
      1% basis = 1
    \else
      \ifnum#1#2=\m@ne % basis = -1
        \ifodd#3#4 %
          -%
        \fi
        1%
      \else % |basis| > 1
        \ifx#3-% power < 0
          0%
        \else % power > 2
          \ifx#1-% basis < 0
            \ifodd#3#4 %
              -%
            \fi
            \InCa@PowRec#2!#3#4!1!%
          \else
            \InCa@PowRec#1#2!#3#4!1!%
          \fi
        \fi
      \fi
    \fi
```

```
  \fi
}
```

```
\InCa@PowRec
```

```
Pow(b, p) {
  PowRec(b, p, 1)
}
PowRec(b, p, r) {
  if p == 1 then
    return r
  else
    ifodd p then
      return PowRec(b*b, p div 2, r*b) % p div 2 = (p-1)/2
    else
      return PowRec(b*b, p div 2, r)
    fi
  fi
}
```

```
\def\InCa@PowRec#1!#2!#3!{%
  \ifnum#2=\@ne
    \ifnum#1>#3 %
      \InCa@Mul#1!#3!%
    \else
      \InCa@Mul#3!#1!%
    \fi
  \else
    \expandafter\InCa@PowRec
    \number\InCa@Mul#1!#1!\expandafter!%
    \number\intcalcShr{#2}\expandafter!%
    \number
    \ifodd#2 %
      \ifnum#1>#3 %
        \InCa@Mul#1!#3!%
      \else
        \InCa@Mul#3!#1!%
      \fi
    \else
      #3%
    \fi
    \expandafter!%
  \fi
}
```

2.5.11 Div

```
\intcalcDiv
```

```
\def\intcalcDiv#1#2{%
  \number\expandafter\InCa@Div
  \number\number#1\expandafter!%
  \number#2! %
}
```

```
\InCa@Div
```

```
\def\InCa@Div#1!#2!{%
  \ifcase#2 %
    0\IntCalcError:DivisionByZero%
  \else
    \ifcase#1 %
      0%
```

```
    \else
      \expandafter\InCa@DivSwitch
      \number#1\expandafter!%
      \number#2!%
    \fi
  \fi
}
```

`\IntCalcDiv`

```
\def\InCa@Temp#1{%
  \def\IntCalcDiv##1!##2!{%
    \number
    \ifcase##2 %
      0\IntCalcError:DivisionByZero%
    \else
      \ifcase##1 %
        0%
      \else
        \InCa@@Div##1!##2!%
      \fi
    \fi
    #1%
  }%
}
\InCa@Temp{ }%
```

`\InCa@DivSwitch` Decision table for `\InCa@DivSwitch`.

$x<0$	$y<0$	$+$	$\mathrm{Div}(-x,-y)$
	else	$-$	$\mathrm{Div}(-x,y)$
else	$y<0$	$-$	$\mathrm{Div}(x,-y)$
	else	$+$	$\mathrm{Div}(x,y)$

```
\def\InCa@DivSwitch#1!#2!{%
  \ifnum#1<\z@
    \ifnum#2<\z@
      \expandafter\InCa@@Div\number-#1\expandafter!%
      \@gobble#2!%
    \else
      -%
      \expandafter\InCa@@Div\@gobble#1!#2!%
    \fi
  \else
    \ifnum#2<\z@
      -%
      \expandafter\InCa@@Div\number#1\expandafter!%
      \@gobble#2!%
    \else
      \InCa@@Div#1!#2!%
    \fi
  \fi
}
```

`\InCa@@Div`

```
\def\InCa@@Div#1!#2!{%
  \ifnum#1>#2 %
    \ifcase#2 % 0 already catched
?     \IntCalcError:ThisCannotHappen%
```

```
    \or % 1
      #1%
    \or % 2
      \InCa@Shr#1!%
    \else
      \InCa@DivStart!#1!#2!#2!%
    \fi
  \else
    \ifnum#1=#2 %
      1%
    \else
      0%
    \fi
  \fi
}
```

\InCa@DivStart

```
\def\InCa@DivStart#1!#2#3!#4#5{%
  \ifx#5!%
    \@ReturnAfterElseFi{%
      \InCa@DivStartI{#1#2}#3=!%
    }%
  \else
    \@ReturnAfterFi{%
      \InCa@DivStart{#1#2}!#3!#5%
    }%
  \fi
}
```

\InCa@StartI

```
\def\InCa@DivStartI#1!#2!{%
  \expandafter\InCa@DivStartII
  \number#2\expandafter\expandafter\expandafter!%
  \intcalcShl{#2}!%
  #1!%
}
```

\InCa@StartII

```
\def\InCa@DivStartII#1!#2!{%
  \expandafter\InCa@DivStartIII
  \number#1\expandafter!%
  \number#2\expandafter\expandafter\expandafter!%
  \intcalcShl{#2}!%
}
```

\InCa@StartIII

```
\def\InCa@DivStartIII#1!#2!#3!{%
  \expandafter\InCa@DivStartIV
  \number#1\expandafter!%
  \number#2\expandafter!%
  \number#3\expandafter!%
  \number\InCa@Add#3!#2!\expandafter\expandafter\expandafter!%
  \intcalcShl{#3}!%
}
```

\InCa@StartIV

```
\def\InCa@DivStartIV#1!#2!#3!#4!#5!#6!{%
  \InCa@ProcessDiv#6!#1!#2!#3!#4!#5!/%
}
```

\InCa@ProcessDiv

```
\def\InCa@ProcessDiv#1#2#3!#4!#5!#6!#7!#8!#9/{%
  #9%
  \ifnum#1<#4 % 0
    0%
    \ifx#2=%
    \else
      \InCa@ProcessDiv{#1#2}#3!#4!#5!#6!#7!#8!%
    \fi
  \else % 1-9
    \ifnum#1<#5 % 1
      1%
      \ifx#2=%
      \else
        \expandafter\InCa@ProcessDiv\expandafter{%
          \number\InCa@Sub#1!#4!%
          #2%
        }#3!#4!#5!#6!#7!#8!%
      \fi
    \else % 2-9
      \ifnum#1<#7 % 2 3 4 5
        \ifnum#1<#6 % 2 3
          \@ReturnAfterElseFi{%
            \expandafter\InCa@@ProcessDiv
            \number\InCa@Sub#1!#5!!%
            23%
          }%
        \else % 4 5
          \@ReturnAfterFi{%
            \expandafter\InCa@@ProcessDiv
            \number\InCa@Sub#1!#6!!%
            45%
          }%
        \fi
        #2#3!#4!#5!#6!#7!#8!%
      \else % 6 7 8 9
        \ifnum#1<#8 % 6 7
          \@ReturnAfterElseFi{%
            \expandafter\InCa@@ProcessDiv
            \number\InCa@Sub#1!#7!!%
            67%
          }%
        \else % 8 9
          \@ReturnAfterFi{%
            \expandafter\InCa@@ProcessDiv
            \number\InCa@Sub#1!#8!!%
            89%
          }%
        \fi
        #2#3!#4!#5!#6!#7!#8!%
      \fi
    \fi
  \fi
  \ifx#2=%
    \expandafter\@gobble
  \fi
  /%
}
```

\InCa@@ProcessDiv

```
\def\InCa@@ProcessDiv#1!#2#3#4#5!#6!{%
  \ifnum#1<#6 %
    #2%
    \@ReturnAfterElseFi{%
      \ifx#4=%
        \expandafter\InCa@CleanupIV
      \else
        \@ReturnAfterFi{%
          \InCa@ProcessDiv{#1#4}#5!#6!%
        }%
      \fi
    }%
  \else
    #3%
    \@ReturnAfterFi{%
      \ifx#4=%
        \expandafter\InCa@CleanupIV
      \else
        \@ReturnAfterFi{%
          \expandafter\InCa@ProcessDiv\expandafter{%
            \number\InCa@Sub#1!#6! %
            #4%
          }#5!#6!%
        }%
      \fi
    }%
  \fi
}
```

\InCa@CleanupIV

```
\def\InCa@CleanupIV#1!#2!#3!#4!{}
```

2.5.12 Mod

\intcalcMod

```
\def\intcalcMod#1#2{%
  \number\expandafter\InCa@Mod
  \number\number#1\expandafter!%
  \number#2! %
}
```

\intcalc@Mod Pseudocode/decision table for `\intcalc@Mod`.

if	$y = 0$	DivisionByZero
elsif	$y < 0$	$-\operatorname{Mod}(-x, -y)$
elsif	$x = 0$	0
elsif	$y = 1$	0
elsif	$y = 2$	$\operatorname{ifodd}(x)\ ?\ 1 : 0$
elsif	$x < 0$	$z \leftarrow x - (x/y) * y; \quad (z < 0)\ ?\ z + y : z$
else		$x - (x/y) * y$

```
\def\InCa@Mod#1!#2!{%
  \ifcase#2 %
    0\IntCalcError:DivisionByZero%
  \else
    \ifnum#2<\z@
```

```
      -%
      \expandafter\InCa@Mod
      \number-#1\expandafter!%
      \number-#2!%
    \else
      \ifcase#1 %
        0%
      \else
        \ifcase#2 % 0 already catched
?         \IntCalcError:ThisCannotHappen%
        \or % 1
          0%
        \or % 2
          \ifodd#1 1\else 0\fi
        \else
          \ifnum#1<\z@
            \expandafter\InCa@ModShift
            \number-%
              \expandafter\InCa@Sub
              \number\@gobble#1\expandafter!%
              \number\intcalcMul{#2}{%
                \expandafter\InCa@Div\@gobble#1!#2!%
              }!%
            !#2!%
          \else
            \expandafter\InCa@Sub\number#1\expandafter!%
            \number\intcalcMul{#2}{\InCa@Div#1!#2!}!%
          \fi
        \fi
      \fi
    \fi
  \fi
}
```

\IntCalcMod

```
\def\InCa@Temp#1{%
  \def\IntCalcMod##1!##2!{%
    \number
    \ifcase##2 %
      0\IntCalcError:DivisionByZero%
    \else
      \ifcase##1 %
        0%
      \else
        \ifcase##2 % 0 already catched
?         \IntCalcError:ThisCannotHappen
        \or % 1
          0%
        \or % 2
          \ifodd ##1 1\else 0\fi
        \else
          \expandafter\InCa@Sub\number##1\expandafter!%
          \number\intcalcMul{##2}{\InCa@Div##1!##2!}!%
        \fi
      \fi
    \fi
    #1%
  }%
```

```
}
\InCa@Temp{ }%
```

`\InCa@ModShift`

```
\def\InCa@ModShift#1!#2!{%
  \ifnum#1<\z@
    \expandafter\InCa@Sub\number#2\expandafter!%
    \@gobble#1!%
  \else
    #1%
  \fi
}
```

2.5.13 Help macros

`\InCa@Empty`

```
\def\InCa@Empty{}
```

`\@gobble`

```
\expandafter\ifx\csname @gobble\endcsname\relax
  \long\def\@gobble#1{}%
\fi
```

`\@ReturnAfterFi`

```
\long\def\@ReturnAfterFi#1\fi{\fi#1}%
```

`\@ReturnAfterElseFi`

```
\long\def\@ReturnAfterElseFi#1\else#2\fi{\fi#1}%
\InCa@AtEnd%
```

⟨/package⟩

3 Installation

3.1 Download

Package. This package is available on CTAN[1]:

CTAN:macros/latex/contrib/intcalc/intcalc.dtx The source file.

CTAN:macros/latex/contrib/intcalc/intcalc.pdf Documentation.

Bundle. All the packages of the bundle 'intcalc' are also available in a TDS compliant ZIP archive. There the packages are already unpacked and the documentation files are generated. The files and directories obey the TDS standard.

CTAN:install/macros/latex/contrib/intcalc.tds.zip

TDS refers to the standard "A Directory Structure for TEX Files" (CTAN:pkg/tds). Directories with `texmf` in their name are usually organized this way.

3.2 Bundle installation

Unpacking. Unpack the `intcalc.tds.zip` in the TDS tree (also known as `texmf` tree) of your choice. Example (linux):

```
unzip intcalc.tds.zip -d ~/texmf
```

[1] CTAN:pkg/intcalc

3.3 Package installation

Unpacking. The `.dtx` file is a self-extracting docstrip archive. The files are extracted by running the `.dtx` through plain TeX:

```
tex intcalc.dtx
```

TDS. Now the different files must be moved into the different directories in your installation TDS tree (also known as `texmf` tree):

```
intcalc.sty → tex/generic/intcalc/intcalc.sty
intcalc.pdf → doc/latex/intcalc/intcalc.pdf
intcalc.dtx → source/latex/intcalc/intcalc.dtx
```

If you have a `docstrip.cfg` that configures and enables docstrip's TDS installing feature, then some files can already be in the right place, see the documentation of docstrip.

3.4 Refresh file name databases

If your TeX distribution (TeX Live, MiKTeX, ...) relies on file name databases, you must refresh these. For example, TeX Live users run `texhash` or `mktexlsr`.

3.5 Some details for the interested

Unpacking with LaTeX. The `.dtx` chooses its action depending on the format:

plain TeX: Run docstrip and extract the files.

LaTeX: Generate the documentation.

If you insist on using LaTeX for docstrip (really, docstrip does not need LaTeX), then inform the autodetect routine about your intention:

```
latex \let\install=y\input{intcalc.dtx}
```

Do not forget to quote the argument according to the demands of your shell.

Generating the documentation. You can use both the `.dtx` or the `.drv` to generate the documentation. The process can be configured by the configuration file `ltxdoc.cfg`. For instance, put this line into this file, if you want to have A4 as paper format:

```
\PassOptionsToClass{a4paper}{article}
```

An example follows how to generate the documentation with pdfLaTeX:

```
pdflatex intcalc.dtx
makeindex -s gind.ist intcalc.idx
pdflatex intcalc.dtx
makeindex -s gind.ist intcalc.idx
pdflatex intcalc.dtx
```

4 History

[2007/09/09 v1.0]

- First version.

[2007/09/27 v1.1]

- `\intcalcNum` added.
- `\intcalcShl` and `\intcalcShr` allow negative numbers. The sign is preserved.
- Reuse `\@gobble` instead of own macro `\IntCalc@Gobble`.
- Small fixes.
- Shorter internal prefix.
- Some programmer's interface.

[2016/05/16 v1.2]

- Documentation updates.

[2019/12/15 v1.3]

- Documentation updates.

5 Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; plain numbers refer to the code lines where the entry is used.

M

N

P

R

T

W

X

Z