




How it works?

1. Enter text and formulas into the "Code" box on the left.
2. Press **F5** or click  to **calculate**. The results will appear in the "Output" box on the right as a professionally formatted Html **report**.
3. Click  to **print** or  to **copy** the output.

You can also **export** it to **Html** , **PDF**  or **MS Word**  document.

The language

The Calcpad language includes the following elements (click an item to insert):

- Real numbers: digits "0" - "9" and decimal point ".";
- Complex numbers: **re** ± **imi** (e.g. **3 - 2i**);
- Variables:
 - Latin letters: "*a*" - "*z*", "*A*" - "*Z*";
 - Greek letters: "*α*" - "*ω*", "*A*" - "*Ω*";
 - digits: "0" - "9";
 - comma: " , ";
 - prime symbols: " ' ", " '' ", " ''' ", " '''' ";
 - special symbols: " ∅ ", " Ø ", " ° ", " ¤ ";
 - "_" for subscript;

A variable name must start with a letter. Names are case sensitive.

- Constants: *π*, *e*, *φ*, *γ*, *g*, *G*, *M_E*, *M_S*, *c*, *h*, *μ₀*, *ε₀*, *k_e*, *e*, *m_e*, *m_p*, *m_n*, *N_A*, *σ*, *k_B*, *R*, *F*, *γ_C*, *γ_S*, *γ_a*, *γ_g*, *γ_w*
- Operators:
 - "!" - factorial;
 - "^" - exponent;
 - "/" - division;
 - "÷" - force division bar;
 - "\" - division;
 - "⊗" - modulo (remainder);
 - "*" - multiplication;
 - "-" - minus;
 - "+" - plus;
 - "≡" - equal to;
 - "≠" - not equal to;
 - "<" - less than;
 - ">" - greater than;
 - "≤" - less or equal;
 - "≥" - greater or equal;
 - "^" - logical "and";
 - "v" - logical "or";

" \oplus " - logical "xor";

"=" - assignment;

- Custom functions of type $f(x; y; z; \dots)$;

- Built-in functions:

- Trigonometric:

sin(x) - sine;

cos(x) - cosine;

tan(x) - tangent;

csc(x) - cosecant;

sec(x) - secant;

cot(x) - cotangent;

- Hyperbolic:

sinh(x) - hyperbolic sine;

cosh(x) - hyperbolic cosine;

tanh(x) - hyperbolic tangent;

csch(x) - hyperbolic cosecant;

sech(x) - hyperbolic secant;

coth(x) - hyperbolic cotangent;

- Inverse trigonometric:

asin(x) - inverse sine;

acos(x) - inverse cosine;

atan(x) - inverse tangent;

atan2(x ; y) - the angle whose tangent is the quotient of y and x ;

acsc(x) - inverse cosecant;

asec(x) - inverse secant;

acot(x) - inverse cotangent;

- Inverse hyperbolic:

asinh(x) - inverse hyperbolic sine;

acosh(x) - inverse hyperbolic cosine;

atanh(x) - inverse hyperbolic tangent;

acsch(x) - inverse hyperbolic cosecant;

asech(x) - inverse hyperbolic secant;

acoth(x) - inverse hyperbolic cotangent;

- Logarithmic, exponential and roots:

log(x) - decimal logarithm;

ln(x) - natural logarithm;

log₂(x) - binary logarithm;

exp(x) – exponential function;

sqr(x) or **sqrt**(x) - square root;

cbrrt(x) - cubic root;

root(x ; n) - n -th root;

◦ Rounding:

round(x) - round to the nearest integer;

floor(x) - round to the lower integer;

ceiling(x) - round to the greater integer;

trunc(x) - round to the nearest integer towards zero;

◦ Integer:

mod(x ; y) - the remainder of an integer division;

gcd(x ; y) - the greatest common divisor of two integers;

lcm(x ; y) - the least common multiple of two integers;

◦ Complex:

abs(x) - absolute value/magnitude;

re(x) - the real part of a complex number;

im(x) - the imaginary part of a complex number;

phase(x) - the phase of a complex number;

◦ Aggregate and interpolation:

min(x ; y ; $z...$) - minimum of multiple values;

max(x ; y ; $z...$) - maximum of multiple values;

sum(x ; y ; $z...$) - sum of multiple values = $x + y + z...$;

sumsq(x ; y ; $z...$) - sum of squares = $x^2 + y^2 + z^2...$;

srss(x ; y ; $z...$) - square root of sum of squares = **sqrt**($x^2 + y^2 + z^2...$);

average(x ; y ; $z...$) - average of multiple values = $(x + y + z...)/n$;

product(x ; y ; $z...$) - product of multiple values = $x \cdot y \cdot z...$;

mean(x ; y ; $z...$) - geometric mean = **n-th root**($x \cdot y \cdot z...$);

take(n ; a ; b ; $c...$) - returns the n -th element from the list;

line(x ; a ; b ; $c...$) - linear interpolation;

spline(x ; a ; b ; $c...$) - Hermite spline interpolation;

◦ Conditional and logical:

if($cond$; $value\text{-if-true}$; $value\text{-if-false}$) - conditional evaluation;

switch($cond1$; $value1$; $cond2$; $value2$; ...; $default$) - selective evaluation;

not(x) - logical "not";

and(x ; y ; $z...$) - logical "and";

or(x ; y ; $z...$) - logical "or";

xor(x ; y ; $z...$) - logical "xor";

◦ Other:

sign(x) - sign of a number;

random(x) - random number between 0 and x .

Comments: "Title" or 'text' in double or single quotes, respectively.

HTML, CSS, JS and SVG are allowed.

- Graphing and plotting:

$\$Plot \{ f(x) @ x = a : b \}$ - simple plot;

$\$Plot \{ x(t) | y(t) @ t = a : b \}$ - parametric;

$\$Plot \{ f_1(x) \& f_2(x) \& \dots @ x = a : b \}$ - multiple;

$\$Plot \{ x_1(t) | y_1(t) \& x_2(t) | y_2(t) \& \dots @ x = a : b \}$ - multiple parametric;

$\$Map \{ f(x; y) @ x = a : b \& y = c : d \}$ - 2D color map of a 3D surface;

PlotHeight - height of plot area in pixels;

PlotWidth - width of plot area in pixels;

PlotStep - grid size for map plotting.

- Iterative and numerical methods:

$\$Root \{ f(x) = const @ x = a : b \}$ - root finding for $f(x) = const$;

$\$Root \{ f(x) @ x = a : b \}$ - root finding for $f(x) = 0$;

$\$Find \{ f(x) @ x = a : b \}$ similar to above, but x is not required to be a precise solution;

$\$Sup \{ f(x) @ x = a : b \}$ - local maximum of a function;

$\$Inf \{ f(x) @ x = a : b \}$ - local minimum of a function;

$\$Area \{ f(x) @ x = a : b \}$ - adaptive Gauss-Lobatto numerical integration;

$\$Integral \{ f(x) @ x = a : b \}$ - Tanh-Sinh numerical integration;

$\$Slope \{ f(x) @ x = a \}$ - numerical differentiation;

$\$Sum \{ f(x) @ k = a : b \}$ - iterative sum;

$\$Product \{ f(k) @ k = a : b \}$ - iterative product;

$\$Repeat \{ f(k) @ k = a : b \}$ - general inline iterative procedure;

Precision - relative precision for numerical methods [10^{-2} ; 10^{-16}] (default is 10^{-12})

- Program flow control:

Simple:

```
#if condition  
  your code goes here  
#end if
```

Alternative:

```
#if condition  
  your code goes here  
#else  
  some other code  
#end if
```

Complete:

```
#if condition1  
  your code goes here
```

```

#else if condition2
    your code goes here
#else
    some other code
#end if

```

You can add or omit as many "#else ifs" as needed. Only one "#else" is allowed.

You can omit this too.

- Iteration blocks:

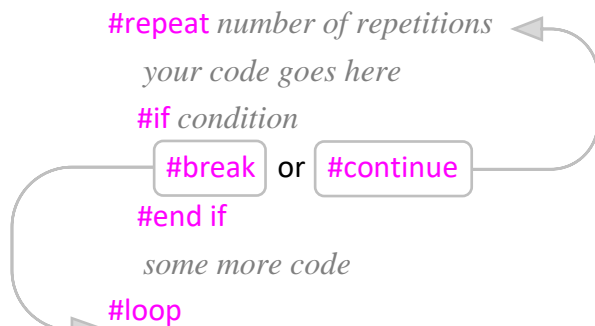
Simple:

```

#repeat number of repetitions
    your code goes here
#loop

```

With conditional break/continue:



- Modules and macros/string variables:

Modules:

```

#include filename - include external file (module);
#local - start local section (not to be included);
#global - start global section (to be included);

```

Inline string variable:

```

#def variable_name$ = content

```

Multiline string variable:

```

#def variable_name$
    content line 1
    content line 2
    ...
#end def

```

Inline macro:

```

#def macro_name$(param1$; param2$; ...) = content

```

Multiline macro:

```

#def macro_name$(param1$; param2$; ...)
    content line 1
    content line 2

```

...
#end def

- Output control:

#hide - hide the report contents;

#show - always show the contents (default);

#pre - show the next contents only before calculations;

#post - show the next contents only after calculations;

#val - show only the final result, without the equation;

#equ - show complete equations and results (default);

#noc - show only equations without results (no calculations);

#round *n* - rounds the output to *n* digits after the decimal point.

- Breakpoints for step-by-step execution:

#pause - calculates down to the current line and waits for the user to resume manually;

#input - renders an input form to the current line and waits for user input.

Each of the above commands is effective after the current line until the end of the report or another command that overwrites it.

- Units for trigonometric functions: **#deg** - degrees, **#rad** - radians, **#gra** - gradians;

- Separator for target units: **|**;

- Return angles with units: *ReturnAngleUnits* = 1;

- Dimensionless: *%*, *‰*;

- Angle: *°*, *'*, *"*, *deg*, *rad*, *grad*, *rev*;

- Metric units (SI and compatible):

Mass: *g*, *hg*, *kg*, *t*, *kt*, *Mt*, *Gt*, *dg*, *cg*, *mg*, *μg*, *Da*, *u*;

Length: *m*, *km*, *dm*, *cm*, *mm*, *μm*, *nm*, *pm*, *AU*, *ly*;

Time: *s*, *ms*, *μs*, *ns*, *ps*, *min*, *h*, *d*;

Frequency: *Hz*, *kHz*, *MHz*, *GHz*, *THz*, *mHz*, *μHz*, *nHz*, *pHz*, *rpm*;

Speed: *kmh*;

Electric current: *A*, *kA*, *MA*, *GA*, *TA*, *mA*, *μA*, *nA*, *pA*;

Temperature: *°C*, *Δ°C*, *K*;

Amount of substance: *mol*;

Luminous intensity: *cd*;

Area: *a*, *daa*, *ha*;

Volume: *L*, *mL*, *cL*, *dL*, *hL*;

Force: *dyn*, *N*, *daN*, *hN*, *kN*, *MN*, *GN*, *TN*, *gf*, *kgf*, *tf*;

Moment: *Nm*, *kNm*;

Pressure: *Pa*, *daPa*, *hPa*, *kPa*, *MPa*, *GPa*, *TPa*, *dPa*, *cPa*, *mPa*, *μPa*, *nPa*, *pPa*,

bar, *mbar*, *μbar*, *atm*, *at*, *Torr*, *mmHg*;

Viscosity: *P*, *cP*, *St*, *cSt*;

Energy work: *J*, *kJ*, *MJ*, *GJ*, *TJ*, *mJ*, *μJ*, *nJ*, *pJ*, *Wh*, *kWh*, *MWh*, *GWh*, *TWh*,

cal, *kcal*, *erg*, *eV*, *keV*, *MeV*, *GeV*, *TeV*, *PeV*, *EeV*;

Power: *W, kW, MW, GW, TW, mW, μW, nW, pW, hpM, ks, VA, kVA, MVA, GVA, TVA, mVA, μVA, nVA, pVA, VAR, kVAR, MVAR, GVAR, TVAR, mVAR, μVAR, nVAR, pVAR*;

Electric charge: *C, kC, MC, GC, TC, mC, μC, nC, pC, Ah, mAh*;

Potential: *V, kV, MV, GV, TV, mV, μV, nV, pV*;

Capacitance: *F, kF, MF, GF, TF, mF, μF, nF, pF*;

Resistance: *Ω, kΩ, MΩ, GΩ, TΩ, mΩ, μΩ, nΩ, pΩ*;

Conductance: *S, kS, MS, GS, TS, mS, μS, nS, pS, Ξ, kΞ, MΞ, GΞ, TΞ, mΞ, μΞ, nΞ, pΞ*;

Magnetic flux: *Wb, kWb, MWb, GWb, TWb, mWb, μWb, nWb, pWb*;

Magnetic flux density: *T, kT, MT, GT, TT, mT, μT, nT, pT*;

Inductance: *H, kH, MH, GH, TH, mH, μH, nH, pH*;

Luminous flux: *lm*;

Illuminance: *lx*;

Radioactivity: *Bq, kBq, MBq, GBq, TBq, mBq, μBq, nBq, pBq, Ci, Rd*;

Absorbed dose: *Gy, kGy, MGy, GGy, TGy, mGy, μGy, nGy, pGy*;

Equivalent dose: *Sv, kSv, MSv, GSv, TSv, mSv, μSv, nSv, pSv*;

Catalytic activity: *kat*;

- Non-metric units (Imperial/US):

Mass: *gr, dr, oz, lb, kip, st, qr, cwt, cwt_uk, cwt_us, ton, ton_uk, ton_us, slug*;

Length: *th, in, ft, yd, ch, fur, mi, ftm, cable, nmi, li, rod, pole, perch, lea*;

Speed: *mph, knot*;

Temperature: *°F, Δ°F, °R*;

Area: *rood, ac*;

Volume (fluid): *fl_oz, gi, pt, qt, gal, bbl, (dry) bu*;

fl_oz_uk, gi_uk, pt_uk, qt_uk, gal_uk, bbl_uk, (dry) bu_uk;

fl_oz_us, gi_us, pt_us, qt_us, gal_us, bbl_us, (dry) bu_us;

Force: *ozf, lbf, kipf, tonf, pdl*;

Pressure: *osi, osf psi, psf, ksi, ksf, tsi, tsf, inHg*;

Energy/work: *BTU, therm, therm_uk, therm_us, quad*;

Power: *hp, hpE, hpS*.