

Как работи?

1. **Въведете** текста и формулите в прозореца "Код" отляво.
 2. Натиснете **F5** или бутона за да **изчислите** резултатите. Те ще се покажат отдясно в прозореца "**Резултати**", като професионално оформена изчислителна **записка**.
 3. Натиснете за да **отпечатате** или за да **копирате** текста на записката.
Може също да го **запишете** като **Html** , **PDF** или **MS Word** документ.

Може също да го **запишете** като Html , PDF или MS Word документ.

Програмен език

Програмният език на **Calcpad** включва следните елементи (кликнете за вмъкване):

- Реални числа: цифри "0" - "9" и десетична точка ".;"
 - Комплексни числа: $re \pm imi$ (например $3 - 2i$);
 - Реални вектори: $[v_1; v_2; v_3; \dots; v_n]$;
 - Реални матрици: $[M_{11}; M_{12}; \dots; M_{1n} | M_{21}; M_{22}; \dots; M_{2n} \dots | M_{m1}; M_{m2}; \dots; M_{mn}]$;
 - Променливи:
 - всякакви букви на Unicode;
 - цифри: 0 - 9;
 - запетая: ",";
 - специални символи: ', " , "'' , "''' , "^-^" , "phi" , "emptyset" , "circ" , "Delta" ;
 - горни индекси: ${}^0, {}^1, {}^2, {}^3, {}^4, {}^5, {}^6, {}^7, {}^8, {}^9, {}^n$, +, -;
 - ДОЛНИ ИНДЕКСИ: ${}_0, {}_1, {}_2, {}_3, {}_4, {}_5, {}_6, {}_7, {}_8, {}_9, {}_+, {}_-, {}_=, (,)$;

Имената на променливи трябва да започват с буква. Различава главни и малки букви.

- Константи:
 π , e , φ , γ , g , G , M_E , M_S , c , h , μ_0 , ε_0 , 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
 - Оператори:
"!" - факториел;
"^" - степенуване;
"/" - делене;
"÷" - делене с дробна черта в линейни и накл. черта във фигури формули (//);
"\\" - целочислено делене;
"%" - остатък (%);
"*" - умножение;
"-" - изваждане;
"+" - събиране;
"≡" - равенство (==);
"≠" - неравенство (!=);
"<" - по-малко;
">" - по-голямо;
"≤" - по-малко или равно (<=);
"≥" - по-голямо или равно (>=);
"∧" - логическо "И"/AND (&&):

- " \vee " - логическо "ИЛИ"/OR (||);
- " \oplus " - изключващо "ИЛИ"/XOR (^);
- " \angle " - фазор A \angle ф (<<);
- " $=$ " - присвояване;
- Потребителски функции от вида $f(x; y; z; \dots)$;
- Вградени функции:

Тригонометрични:

$\sin(x)$	- синус;
$\cos(x)$	- косинус;
$\tan(x)$	- тангенс;
$\csc(x)$	- косеканс;
$\sec(x)$	- секанс;
$\cot(x)$	- котангенс;

Хиперболични:

$\sinh(x)$	- синус хиперболичен;
$\cosh(x)$	- косинус хиперболичен;
$\tanh(x)$	- тангенс хиперболичен;
$\csch(x)$	- косеканс;
$\sech(x)$	- секанс;

Обратни тригонометрични:

$\coth(x)$	- котангенс хиперболичен;
$\asin(x)$	- аркуссинус;
$\acos(x)$	- аркускосинус;
$\atan(x)$	- аркустангенс;
$\atan2(x; y)$	- ъгъл, чиито тангенс е отношението на y към x ;
$\acsc(x)$	- аркускосеканс;
$\asec(x)$	- аркуссеканс;
$\acot(x)$	- аркускотангенс;

Обратни хиперболични:

$\sinh(x)$	- аркуссинус хиперболичен;
$\cosh(x)$	- аркускосинус хиперболичен;
$\tanh(x)$	- аркустангенс хиперболичен;
$\csch(x)$	- аркускосеканс хиперболичен;
$\sech(x)$	- аркуссеканс хиперболичен;
$\coth(x)$	- аркускотангенс хиперболичен;

Логаритмични, експоненциални и корени:

$\log(x)$	- десетичен логаритъм;
$\ln(x)$	- натурален логаритъм;
$\log_2(x)$	- двоичен логаритъм;
$\exp(x)$	- експоненциална функция;

sqr(x) или **sqrt(x)** - корен квадратен;

cbrt(x) - корен кубичен;

root(x; n) - корен n-ти;

Закръгляване:

round(x) - закръгляване до най-близкото цяло число;

floor(x) - закръгляване до по-малкото цяло число;

ceiling(x) - закръгляване до по-голямото цяло число;

trunc(x) - закръгляване към по-близкото число в посока към нулата;

Целочислени:

mod(x, y) - остатък от деление;

gcd(x; y; z...) - най-голям общ делител;

lcm(x; y; z...) - най-малко общо кратно;

Комплексни:

abs(x) - абсолютна стойност;

re(x) - реалната част на комплексно число;

im(x) - имагинерната част на комплексно число;

phase(x) - фаза на комплексно число;

conj(z) - спрегнато комплексно число;

Агрегатни и интерполяционни:

min(x; y; z...) - минимум на множество стойности;

max(x; y; z...) - максимум на множество стойности;

sum(x; y; z...) - сума на множество стойности = $x + y + z \dots$;

sumsq(x; y; z...) - сума от квадратите = $x^2 + y^2 + z^2 \dots$;

srss(x; y; z...) - корен квадратен от сумата на квадратите = $\sqrt{x^2 + y^2 + z^2 \dots}$;

average(x; y; z...) - средно аритметично от множество стойности = $(x + y + z \dots) / n$;

product(x; y; z...) - произведение на множество стойности = $x \cdot y \cdot z \dots$;

mean(x; y; z...) - средно геометрично = n-th **root(x·y·z...)**;

take(x; a; b; c; ...) - връща n-тия елемент от списъка;

line(x; a; b; c; ...) - линейна интерполяция;

spline(x; a; b; c; ...) - spline интерполяция на Ермит;

Условни и логически:

if(условие; стойност-при-истина; стойност-при-неистина) - условно изчисление;

switch(усл1; стойност1; усл2; стойност2; ... ; по-подразб.) - избирателно изчисление;

not(x) - логическо отрицание (NOT);

and(x; y; z...) - логическо "И" (AND);

or(x; y; z...) - логическо "ИЛИ" (OR);

xor(x; y; z...) - изключващо "ИЛИ" (XOR);

Други:

sign(x) - знак на число;

random(x) - произволно число между 0 и x ;

getunits(x) - връща мерните единици на x без числото или 1 ако x няма мерни единици;

setunits($x; u$) - задава мерни единици u на x , където x може да бъде скалар, вектор или матрица;

clrunits(x) - изчиства мерните единици от скалар, вектор или матрица x ;

hp(x) - преобразува x към еквивалентния му високопроизводителен (hp) тип;

ishp(x) - проверява дали типа на x е високопроизводителен (hp) вектор или матрица;

Векторни:

Създаване и инициализация:

vector(n) - създава празен вектор с дължина n ;

vector_hp(n) - създава празен високопроизводителен (hp) вектор с дължина n ;

range($x_1; x_n; s$) - създава вектор от стойностите в интервала от x_1 до x_n със стъпка s ;

range_hp($x_1; x_n; s$) - създава високопроизводителен (hp) вектор от стойностите в посочения интервал;

Структурни:

len(\vec{v}) - връща дължината на вектора \vec{v} ;

size(\vec{v}) - действителния размер на вектора \vec{v} (индекса на последния ненулев елемент);

resize($\vec{v}; n$) - задава нова дължина n на вектора \vec{v} ;

fill($\vec{v}; x$) - запълва вектора \vec{v} със стойност x ;

join($A; \vec{b}; c\dots$) - създава вектор чрез обединяване на аргументите в списъка - матрици, вектори и скалари;

slice($\vec{v}; i_1; i_2$) - връща частта от вектора \vec{v} , ограничена от индекси i_1 и i_2 , вкл.;

first($\vec{v}; n$) - първите n елемента на вектора \vec{v} ;

last($\vec{v}; n$) - последните n елемента на вектора \vec{v} ;

extract($\vec{v}; \vec{i}$) - извлича онези елементи от \vec{v} , чиито индекси се съдържат в \vec{i} ;

Данни:

sort(\vec{v}) - сортира вектора \vec{v} във възходящ ред;

rsort(\vec{v}) - сортира вектора \vec{v} в низходящ ред;

order(\vec{v}) - индексите на \vec{v} , подредени по възходящ ред на неговите елементи;

revorder(\vec{v}) - индексите на \vec{v} , подредени по низходящ ред на неговите елементи;

reverse(\vec{v}) - нов вектор, съдържащ елементите на \vec{v} в обратен ред;

count($\vec{v}; x; i$) - броя на елементите в \vec{v} , от i -тия нататък, които са равни на x ;

search($\vec{v}; x; i$) - индекса на първия елемент в \vec{v} , от i -тия нататък, който е равен на x ;

find($\vec{v}; x; i$) или

find_eq($\vec{v}; x; i$) - индексите на всички елементи в \vec{v} , от i -тия нататък, които са $= x$;

find_ne($\vec{v}; x; i$) - индексите на всички елементи в \vec{v} , от i -тия нататък, които са $\neq x$;

find_lt($\vec{v}; x; i$) - индексите на всички елементи в \vec{v} , от i -тия нататък, които са $< x$;

find_le($\vec{v}; x; i$) - индексите на всички елементи в \vec{v} , от i -тия нататък, които са $\leq x$;

find_gt($\vec{v}; x; i$) - индексите на всички елементи в \vec{v} , от i -тия нататък, които са $> x$;

find_ge(\vec{v} ; x ; i) - индексите на всички елементи в \vec{v} , от i -тия нататък, които са $\geq x$;

lookup(\vec{a} ; \vec{b} ; x) или

lookup_eq(\vec{a} ; \vec{b} ; x) - всички елементи в \vec{a} , за които съответните елементи в \vec{b} са $= x$;

lookup_ne(\vec{a} ; \vec{b} ; x) - всички елементи в \vec{a} , за които съответните елементи в \vec{b} са $\neq x$;

lookup_lt(\vec{a} ; \vec{b} ; x) - всички елементи в \vec{a} , за които съответните елементи в \vec{b} са $< x$;

lookup_le(\vec{a} ; \vec{b} ; x) - всички елементи в \vec{a} , за които съответните елементи в \vec{b} са $\leq x$;

lookup_gt(\vec{a} ; \vec{b} ; x) - всички елементи в \vec{a} , за които съответните елементи в \vec{b} са $> x$;

lookup_ge(\vec{a} ; \vec{b} ; x) - всички елементи в \vec{a} , за които съответните елементи в \vec{b} са $\geq x$;

Математически:

norm_1(\vec{v}) - L1 (Манхатън) норма на вектора \vec{v} ;

norm(\vec{v}) или

norm_2(\vec{v}) или

norm_e(\vec{v}) - L2 (Евклидова) норма на вектора \vec{v} ;

norm_p(\vec{v} ; p) - L_p норма на вектора \vec{v} ;

norm_i(\vec{v}) - L_∞ (безкрайна) норма на вектора \vec{v} ;

unit(\vec{v}) - нормализирания (единичен) вектор \vec{v} (с L2 норма = 1);

dot(\vec{a} ; \vec{b}) - скаларно произведение на два вектора \vec{a} и \vec{b} ;

cross(\vec{a} ; \vec{b}) - векторно произведение на два вектора \vec{a} и \vec{b} (с дължина 2 или 3);

Матрични:

Създаване и инициализация:

matrix(m ; n) - създава празна матрица с размери $m \times n$;

identity(n) - създава единична матрица с размери $n \times n$;

diagonal(n ; d) - създава диагонална $n \times n$ матрица и запълва главния диагонал със стойност d ;

column(m ; c) - създава матрица-стълб с размери $m \times 1$, запълнена със стойност c ;

utriang(n) - създава горна триъгълна матрица с размери $n \times n$;

ltriang(n) - създава долната триъгълна матрица с размери $n \times n$;

symmetric(n) - създава симетрична матрица с размери $n \times n$;

matrix_hp(m ; n) - създава празна високопроизводителна (hp) матрица с размери $m \times n$;

identity_hp(n) - създава единична високопроизв. (hp) матрица с размери $n \times n$;

diagonal_hp(n ; d) - създава високопроизв. (hp) диагонална $n \times n$ матрица и запълва диагонала със стойност d ;

column_hp(m ; c) - създава високопроизводителна (hp) матрица-стълб с размери $m \times 1$, запълнена със стойност c ;

utriang_hp(n) - създава високопроизв. (hp) горна триъгълна матрица с размери $n \times n$;

ltriang_hp(n) - създава високопроизв. (hp) долната триъгълна матрица с размери $n \times n$;

symmetric_hp(n) - създава симетрична матрица с размери $n \times n$;

vec2diag(\vec{v}) - създава диагонална матрица от елементите на вектора \vec{v} ;

vec2row(\vec{v}) - създава матрица-ред от елементите на вектора \vec{v} ;

vec2col(\vec{v}) - създава матрица-стълб от елементите на вектора \vec{v} ;

- join_cols**($\vec{c}_1; \vec{c}_2; \vec{c}_3\dots$) - създава нова матрица чрез обединяване на вектори в стълбове;
- join_rows**($\vec{r}_1; \vec{r}_2; \vec{r}_3\dots$) - създава нова матрица чрез обединяване на вектори в редове;
- augment**($A; B; C\dots$) - създава нова матрица чрез присъединяване на матриците $A, B, C\dots$ една до друга;
- stack**($A; B; C\dots$) - създава нова матрица чрез присъединяване на матриците $A, B, C\dots$ една под друга;

Структурни:

- n_rows**(M) - броя на редовете в матрицата M ;
- n_cols**(M) - броя на стълбовете в матрицата M ;
- resize**($M; m; n$) - задава нови размери m и n на матрицата M ;
- fill**($M; x$) - запълва матрицата M със стойност x ;
- fill_row**($M; i; x$) - запълва i -тия ред на матрицата M със стойност x ;
- fill_col**($M; j; x$) - запълва j -тия стълб на матрицата M със стойност x ;
- copy**($A; B; i; j$) - копира всички елементи от A в B , започвайки от индекси i и j на B ;
- add**($A; B; i; j$) - добавя всички елементи от A към тези на B , започвайки от индекси i и j на B ;
- row**($M; i$) - извлича i -тия ред на матрицата M като вектор;
- col**($M; j$) - извлича j -тия стълб на матрицата M като вектор;
- extract_rows**($M; \vec{i}$) - извлича онези редове от матрицата M чиито индекси се съдържат във вектор \vec{i} ;
- extract_cols**($M; \vec{j}$) - извлича онези стълбове от матрицата M чиито индекси се съдържат във вектор \vec{j} ;
- diag2vec**(M) - извлича диагоналните елементи от матрицата M като вектор;
- submatrix**($M; i_1; i_2; j_1; j_2$) - извлича подматрица на M , ограничена от редове i_1 и i_2 и стълбове j_1 и j_2 , вкл.;

Данни:

- sort_cols**($M; i$) - сортира стълбовете на M на базата на стойностите в ред i във възходящ ред;
- rsort_cols**($M; i$) - сортира стълбовете на M на базата на стойностите в ред i в низходящ ред;
- sort_rows**($M; j$) - сортира редовете на M на базата на стойностите в стълб j във възходящ ред;
- rsort_rows**($M; j$) - сортира редовете на M на базата на стойностите в стълб j в низходящ ред;
- order_cols**($M; i$) - индексите на стълбовете на M , подредени възходящо по стойностите от ред i ;
- revorder_cols**($M; i$) - индексите на стълбовете на M , подредени низходящо по стойностите от ред i ;
- order_rows**($M; j$) - индексите на редовете на M , подредени възходящо по стойностите от стълб j ;
- revorder_rows**($M; j$) - индексите на редовете на M , подредени низходящо по

стойностите от стълб j ;

mcount($M; x$) - броя на елементите със стойност x в матрицата M ;

msearch($M; x; i; j$) - вектор с двата индекса на първия елемент със стойност x в матрицата M , започвайки от индекси i и j ;

mfind($M; x$) или

mfind_eq($M; x$) - индексите на всички елементи в M , които са $= x$;

mfind_ne($M; x$) - индексите на всички елементи в M , които са $\neq x$;

mfind_lt($M; x$) - индексите на всички елементи в M , които са $< x$;

mfind_le($M; x$) - индексите на всички елементи в M , които са $\leq x$;

mfind_gt($M; x$) - индексите на всички елементи в M , които са $> x$;

mfind_ge($M; x$) - индексите на всички елементи в M , които са $\geq x$;

hlookup($M; x; i_1; i_2$) или

hlookup_eq($M; x; i_1; i_2$) - стойностите от ред i_2 на M , за които елементите от ред i_1 са $= x$;

hlookup_ne($M; x; i_1; i_2$) - стойностите от ред i_2 на M , за които елементите от ред i_1 са $\neq x$;

hlookup_lt($M; x; i_1; i_2$) - стойностите от ред i_2 на M , за които елементите от ред i_1 са $< x$;

hlookup_le($M; x; i_1; i_2$) - стойностите от ред i_2 на M , за които елементите от ред i_1 са $\leq x$;

hlookup_gt($M; x; i_1; i_2$) - стойностите от ред i_2 на M , за които елементите от ред i_1 са $> x$;

hlookup_ge($M; x; i_1; i_2$) - стойностите от ред i_2 на M , за които елементите от ред i_1 са $\geq x$;

vlookup($M; x; j_1; j_2$) или

vlookup_eq($M; x; j_1; j_2$) - стойностите от стълб j_2 на M , за които елементите от стълб j_1 са $= x$;

vlookup_ne($M; x; j_1; j_2$) - стойностите от стълб j_2 на M , за които елементите от стълб j_1 са $\neq x$;

vlookup_lt($M; x; j_1; j_2$) - стойностите от стълб j_2 на M , за които елементите от стълб j_1 са $< x$;

vlookup_le($M; x; j_1; j_2$) - стойностите от стълб j_2 на M , за които елементите от стълб j_1 са $\leq x$;

vlookup_gt($M; x; j_1; j_2$) - стойностите от стълб j_2 на M , за които елементите от стълб j_1 са $> x$;

vlookup_ge($M; x; j_1; j_2$) - стойностите от стълб j_2 на M , за които елементите от стълб j_1 са $\geq x$;

Математически:

hprod($A; B$) - произведение на Hadamard на матриците A и B ;

fprod($A; B$) - произведение на Frobenius на матриците A и B ;

kprod($A; B$) - произведение на Kronecker на матриците A и B ;

mnorm_1(M) - L1 норма на матрицата M ;

mnorm(M) или

mnorm_2(M)	- L2 норма на матрицата M ;
mnorm_e(M)	- норма на Frobenius на матрицата M ;
mnorm_i(M)	- L^∞ норма на матрицата M ;
cond_1(M)	- число на обусловеност на M на база на L1 нормата;
cond(M) или cond_2(M)	- число на обусловеност на M на база на L2 нормата;
cond_e(M)	- число на обусловеност на M на база на нормата на Frobenius;
cond_i(M)	- число на обусловеност на M на база на L^∞ нормата;
det(M)	- детерминанта на матрицата M ;
rank(M)	- ранг на матрицата M ;
trace(M)	- следа на матрицата M ;
transp(M)	- транспонираната матрица на M ;
adj(M)	- адюнгираната матрица на M ;
cofactor(M)	- кофакторната матрица на M ;
eigenvals(M)	- собствените стойности на матрицата M ;
eigenvecs(M)	- собствените вектори на матрицата M ;
cholesky(M)	- декомпозиция на Холецки на симетрична, положително определена матрица M ;
lu(M)	- LU декомпозиция на матрицата M ;
qr(M)	- QR декомпозиция на матрицата M ;
svd(M)	- декомпозиция по особени стойности на M ;
inverse(M)	- обратната матрица на M ;
lsolve($A; \vec{b}$)	- решава системата линейни уравнения $A\vec{x} = \vec{b}$ чрез LDLT декомпозиция за симетрични матрици и LU декомпозиция за несиметрични;
clsolve($A; \vec{b}$)	- решава системата линейни уравнения $A\vec{x} = \vec{b}$ със симетрична, положително определена матрица на коефициентите A посредством декомпозиция на Холецки;
slsolve($A; \vec{b}$)	- решава системата линейни уравнения $A\vec{x} = \vec{b}$ с високопроизводителна симетрична, положително определена матрица на коефициентите A чрез метода на преобусловения спрегнат градиент (PCG);
msolve($A; B$)	- решава обобщеното матрично уравнение $AX = B$ чрез LDLT деком- позиция за симетрични матрици и LU декомпозиция за несиметрични;
cmsolve($A; B$)	- решава обобщеното матрично уравнение $AX = B$ със симетрична, положително определена матрица на коефициентите A посредством декомпозиция на Холецки;
smsolve($A; B$)	- решава обобщеното матрично уравнение $AX = B$ с високопроизводителна симетрична, положително определена матрица на коефициентите A чрез метода на преобусловения спрегнат градиент (PCG);

matmul ($A; B$) - бързо умножение на квадратни hp матрици чрез паралелен алгоритъм на Виноград. Операторът за умножение $A * B$ използва тази функция вътрешно за всички квадратни hp матрици с размер 10 или по-големи;

Двойна интерполяция:

- | | |
|-----------------------------|--|
| take ($x; y; M$) | - връща елемента на матрицата M с индекси x и y ; |
| line ($x; y; M$) | - двойна линейна интерполяция от елементите на M на база на стойностите на x и y ; |
| spline ($x; y; M$) | - двойна spline интерполяция на Ермит от елементите на матрицата M на база на стойностите на x и y . |

Коментари: "Заглавие" или 'текст', съответно в двойни и единични кавички. Разрешено е използването на **HTML**, **CSS**, **JS** и **SVG** в коментарите.

- Графики на функции:

- | | |
|--|--|
| \$Plot{f(x) @ x = a : b} | - стандартна, единична; |
| \$Plot{x(t) y(t) @ t = a : b} | - параметрична; |
| \$Plot{f1(x) & f2(x) & ... @ x = a : b} | - паралелни; |
| \$Plot{x1(t) y1(t) & x2(t) y2(t) & ... @ t = a : b} | - паралелни параметрични; |
| \$Map{f(x; y) @ x = a : b & y = c : d} | - изохроми на 2D функция; |
| PlotHeight | - височина на полето за чертане в пиксели; |
| PlotWidth | - ширина на полето за чертане в пиксели; |
| PlotSVG | - чертай графиките във векторен (SVG) формат; |
| PlotAdaptive | - използвай адаптивно сгъстяване на мрежата за графики на функции; |
| PlotStep | - стъпка на мрежата за интерполиране; |
| PlotShadows | - чертай повърхнините със светлосенки; |
| PlotLightDir | - посока към източника на светлина (0-7); |
| PlotPalette | - номер на палитра (0-9); |
| PlotSmooth | - плавно преливане на цветовете (= 1) или изохроми (= 0) ; |

- Итеративни и числени методи:

- | | |
|---|---|
| \$Root{f(x) = const @ x = a : b} | - намиране на корен на $f(x) = \text{const}$; |
| \$Root{f(x) @ x = a : b} | - намиране на корен на $f(x) = 0$; |
| \$Find{f(x) @ x = a : b} | - намира мястото, където функцията пресича абсцисата, но не се изисква стриктно x да е решение; |
| \$Sup{f(x) @ x = a : b} | - локален максимум на функция; |
| \$Inf{f(x) @ x = a : b} | - локален минимум на функция; |
| \$Area{f(x) @ x = a : b} | - числено интегриране с адаптивна квадратура на Гаус-Лобато-Кронрод; |
| \$Integral{f(x) @ x = a : b} | - числено интегриране с Tanh-Sinh квадратура; |
| \$Slope{f(x) @ x = a : b} | - числено диференциране; |
| \$Sum{f(x) @ x = a : b} | - крайна сума; |
| \$Product{f(k) @ k = a : b} | - крайно произведение; |

<code>\$Repeat{f(k) @ k = a : b}</code>	- итеративен блок от изрази с брояч;
<code>\$While{условие; изрази }</code>	- итеративен блок от изрази с условие;
<code>\$Block{ изрази }</code>	- многоредов блок от изрази;
<code>\$Inline{ изрази }</code>	- едноредов блок от изрази;

Precision - точност за числени методи [10⁻²; 10⁻¹⁶] (по подразбиране - 10⁻¹⁴);

- Условни разклонения:

Стандартно:

```
#if условие
    тук въведете код
#end if
```

Алтернативно:

```
#if условие
    тук въведете код
#else
    алтернативен код
#endif if
```

Пълно:

```
#if условие1
    тук въведете код
#else if условие2
    тук въведете код
#else
    алтернативен код
#endif if
```

Може да добавяте произволен брой "#else if" блокове, но само един "#else".

- Блок за цикъл:

Стандартен:

```
#repeat брой повторения
    тук въведете код
#loop
```

Условен:

```
#repeat брой повторения
    тук въведете код
    #if условие
        #break или #continue
    #end if
    още код
#loop
```

С брояч:

```
#for counter = start : end
    тук въведете код
#loop
```

С условие:

```
#while условие
    тук въведете код
#loop
```

- Модули и макроси/текстови променливи:

Модули:

```
#include име_на_файл - вмъква код от външен файл (модул);
#local - начало на локална секция (не се вмъква);
#global - начало на глобална секция (вмъква се);
```

Едноредова текстова променлива:

```
#def variable_name$ = съдържание
```

Многоредова текстова променлива:

```
#def variable_name$
    съдържание ред 1
    съдържание ред 2
    ...
#end def
```

Едноредов макрос:

```
#def macro_name$(param1$; param2$; ...) = съдържание
```

Многоредов макрос:

```
#def macro_name$(param1$; param2$; ...)
    съдържание ред 1
    съдържание ред 2
    ...
#end def
```

Текстови/CSV файлове:

```
#read M from filename.txt@R1C1:R2C2 TYPE=R SEP=',' - четене на матрица M от текстов/CSV файл;
```

```
#write M to filename.txt@R1C1:R2C2 TYPE=N SEP=',' - запис на матрица M в текстов/CSV файл;
```

```
#append M to filename.txt@R1C1:R2C2 TYPE=N SEP=',' - добавя матрица M към текстов/CSV файл;
```

Excel файлове (xlsx и xlsm):

```
#read M from filename.xlsx@Sheet1!A1:B2 TYPE=R - четене на матрица M от Excel файл;
```

```
#write M to filename.xlsx@Sheet1!A1:B2 TYPE=N - запис на матрица M в Excel файл;
```

```
#append M to filename.xlsx@Sheet1!A1:B2 TYPE=N - добавяне на матрица M към Excel файл;
```

Sheet, range, TYPE и SEP могат да бъдат пропуснати.

За командата #read, TYPE може да бъде някое от [R|D|C|S|U|L|V].

За командите #write и #append, TYPE може да бъде Y или N.

- Контрол на видимостта:

#hide	- скривай съдържанието на документа;
#show	- показвай винаги съдържанието (по подразбиране);
#pre	- показвай следващото съдържание само при въвеждане;
#post	- показвай следващото съдържание само в резултатите;
#val	- показвай само изчислените стойности;

#equ - показвай пълните формули (по подразбиране);
#noc - показвай само формули без стойности (no calculations);
#nosub - не замествай стойностите на променливите (no substitution);
#novar - показвай само заместените стойности на променливите (no variables);
#varsub - показвай формулите с променливи и заместени стойности (по подразбиране);
#split - разделяй уравнения, които не се събират на един ред;
#wrap - свивай уравнения които не се събират на един ред (по подразбиране);
#round *n* - закръглява изходните стойности до *n* цифри след десетичната точка;
#round default - възстановява закръгяването по подразбиране;
#format FFFF - задава потребителски форматиращ низ;
#format default - възстановява форматирането по подразбиране;
#md on - включва използването на markdown в коментари;
#md off - изключва използването на markdown в коментари;
#phasor - задава изходното форматиране на комплексни числа като полярен фазор: $A\angle\phi$;
#complex - задава изходното форматиране на комплексни числа в алгебричен формат: $a + bi$.

Всяка от горните команди е валидна от мястото на използването и до края на документа или докато не бъде отменена от алтернативна команда.

- Точки на прекъсване (постъпково изпълнение):

#pause - изчислява до съответния ред и спира на пауза;
#input - показва формуляр за вход на данни до съответния ред и спира на пауза.

- Единици за тригонометрични функции: **#deg** - градуси, **#rad** - радиани, **#gra** - гради;
- Разделител за отправни единици: **|**;
- Връщай резултати от тригонометр. функции с мерни единици: *ReturnAngleUnits* = 1;
- Бездименсионни единици: **%**, **‰**, **‰‰**, **pcm**, **ppm**, **ppb**, **ppt**, **ppq**;
- Единици за ъгли: **°**, **'**, **"**, **deg**, **rad**, **grad**, **rev**;
- Метрични единици (SI и съвместими):

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

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

Време: **s**, **ms**, **µs**, **ns**, **ps**, **min**, **h**, **d**, **w**, **y**;

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

Скорост: **kmh**;

Електрически поток: **A**, **kA**, **MA**, **GA**, **TA**, **mA**, **µA**, **nA**, **pA**;

Температура: **°C**, **Δ°C**, **K**;

Количество вещества: **mol**;

Интензитет на светлината: **cd**;

Площ: **a**, **daa**, **ha**;

Обем: **L**, **daL**, **hL**, **dL**, **cL**, **mL**, **µL**, **nL**, **pL**;

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

Момент: **Nm**, **kNm**;

Налягане: **Pa**, **daPa**, **hPa**, **kPa**, **MPa**, **GPa**, **TPa**,

dPa, **cPa**, **mPa**, **µPa**, **nPa**, **pPa**,

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

- Вискозитет: **P, cP, St, cSt;**
 Енергия/работка: **J, kJ, MJ, GJ, TJ, mJ, μJ, nJ, pJ,**
Wh, kWh, MWh, GWh, TWh, cal, kcal, erg,
eV, keV, MeV, GeV, TeV, PeV, EeV;
- Мощност: **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;
- Електрически заряд: **C, kC, MC, GC, TC, mC, μC, nC, pC, Ah, mAh;**
 Напрежение: **V, kV, MV, GV, TV, mV, μV, nV, pV;**
- Капацитет: **F, kF, MF, GF, TF, mF, μF, nF, pF;**
 Съпротивление: **Ω, kΩ, MΩ, GΩ, TΩ, mΩ, μΩ, nΩ, pΩ;**
- Проводимост: **S, kS, MS, GS, TS, mS, μS, nS, pS,**
℧, k℧, M℧, G℧, T℧, m℧, μ℧, n℧, p℧;
- Магнитен поток: **Wb, kWb, MWb, GWb, TWb, mWb, μWb, nWb, pWb;**
 Пътност на потока: **T, kT, MT, GT, TT, mT, μT, nT, pT;**
 Индукция: **H, kH, MH, GH, TH, mH, μH, nH, pH;**
- Светлинен поток: **lm;**
 Осветеност: **lx;**
 Радиоактивност: **Bq, kBq, MBq, GBq, TBq, mBq, μBq, nBq, pBq, Ci, Rd;**
 Погълната доза: **Gy, kGy, MGy, GGy, TGy, mGy, μGy, nGy, pGy;**
 Еквивалентна доза: **Sv, kSv, MSv, GSv, TSv, mSv, μSv, nSv, pSv;**
 Активност на катализатор: **kat;**
- Неметрични единици (UK/US):

Маса: **gr, dr, oz, lb** (или **lbf, lb_m**), **kipm** (или **kip_m**), **st, qr,**
cwt (или **cwt_uk, cwt_us**), **ton** (или **ton_uk, ton_us**), **slug;**
 Дължина: **th, in, ft, yd, ch, fur, mi, ftm** (или **ftm_uk, ftm_us**),
cable (или **cable_uk, cable_us**), **nmi, li, rod, pole, perch, lea;**
 Скорост: **mph, knot;**
 Температура: **°F, Δ°F, °R;**
 Площ: **rood, ac;**
 Обем, течност: **fl_oz, gi, pt, qt, gal, bbl, or:**
fl_oz_uk, gi_uk, pt_uk, qt_uk, gal_uk, bbl_uk,
fl_oz_us, gi_us, pt_us, qt_us, gal_us, bbl_us;
 Обем, сух: **(US) pt_dry, (US) qt_dry, (US) gal_dry, (US) bbl_dry,**
pk (или **pk_uk, pk_us**), **bu** (или **bu_uk, bu_us**);
 Сила: **ozf** (или **oz_f**), **lbf** (или **lb_f**), **kip** (или **kipf, kip_f**), **tonf** (или **ton_f**), **ndl;**
 Налягане: **osi, osf, psi, psf, ksi, ksf, tsi, tsf, inHg;**
 Енергия/работка: **BTU, therm** (или **therm_uk, therm_us**), **quad;**
 Мощност: **hp, hpE, hpS;**
 - Потребителски единици: **.Име** = израз.
 Имената могат да съдържат и символи за валута: **€, ₣, ₩, ₪, ₧, ₩, ₩.**

Готови оразмерителни програми по Еврокод

Разполагаме с богата библиотека от оразмерителни програми за Calcpad по Еврокод, които може да ползвате в готов вид, на символични цени.

Пълен списък от разработените програми, ще намерите на следния линк:

<https://www.proektsoft.bg/calcpad/Pricelist-2025-Calcpad.pdf>

Как да поръчаме?

1. Изберете програмите, които са Ви необходими.
2. Изпратете ни списък с номерата на избраните записи или пакети по имейл.
3. Ще Ви подгответим и изпратим индивидуална оферта.

За заявки, пишете на:

proektsoft.bg@gmail.com

Заплащането е еднократно, без абонамент. Веднъж закупени, програмите могат да се ползват без ограничение.