ПРОЕКТСОФТ ®
https://proektsoft.bg

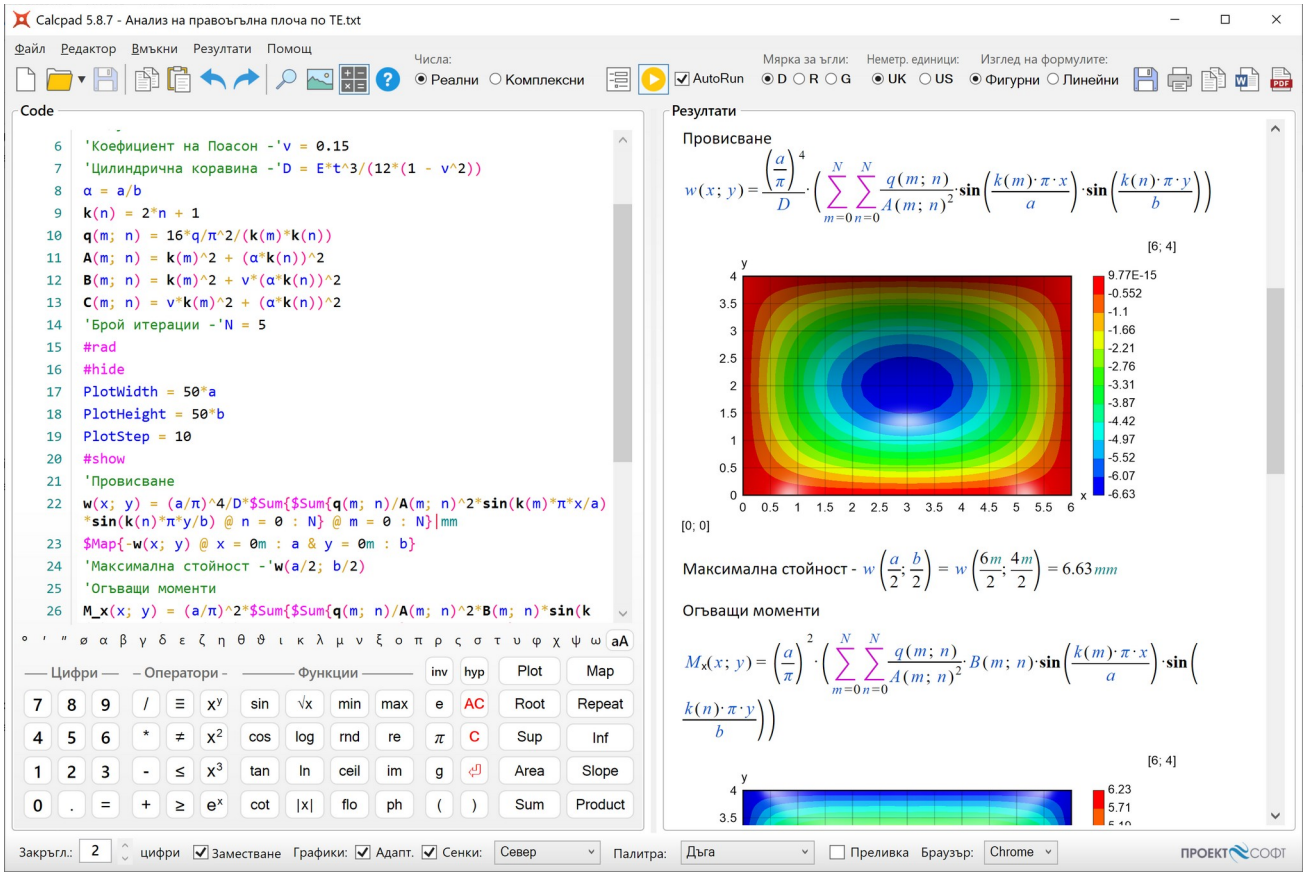
София 1164, бул Пейо Яворов 34-36
+359 2 423 4455
proektsoft.bg@gmail.com

Ново в Calcpad версия 5.8.7

Скъпи приятели,

Версия 5.8.7. на българския математически софтуер с отворен код Calcpad, вече е факт. Тя е резултат, както от усилията на специалистите от Проектсофт, така и на подкрепата на все още малката, но и много активна и сплотена общност в GitHub.

Спрямо предишната версия 5.7.1, новата съдържа множество подобрения и нови функции, които сме описали по-долу. Изчислителното ядро е оптимизирано значително, като подобрението в производителността е около **два пъти**.



Може да изтеглите новата версия от линка:

https://www.proektsoft.bg/calcpad/calcpad-setup-x64.zip

За да работите, е необходимо да имате 64-битов компютър с Windows 7 – 11 и инсталиран Microsoft .NET 6.0, който може да изтеглите оттук:

https://download.visualstudio.microsoft.com/download/pr/fe8415d4-8a35-4af9-80a5-51306a96282d/05f9b2a1b4884238e69468e49b3a5453/windowsdesktop-runtime-6.0.9-win-x64.exe

Ето и подробно описание на новите възможности:

Модули

Ако имате части от кода, които често се повтарят, може да ги организирате в модули и да ги преизползвате многократно. Също така, ако имате по-дълга записка, може да я разделите в няколко файла за по-лесна поддръжка и обновяване. След това, може да ги включите в основния файл, като използвате следния израз:

```
#include име_на_файл
```

Името на файла трябва да съдържа пълния път до локален за компютъра файл. По изключение, ако файлът се намира в същата директория, може да посочите само името.

Във всеки модул, може да определяте кои части от съдържанието подлежат на включване и кои не, като ги дефинирате като локални и глобални блокове, както следва:

`#local` – начало на локален блок от съдържание (код);

`#global` – край на локален блок и начало на глобален.

Ако нищо не е посочено, по подразбиране цялото съдържание се приема за глобално и ще бъде вмъкнато при команда #include.

Възможни са множество нива на множество нива на включване на модули. Това означава, че включения модул може на свой ред да включва други в себе си и т.н.

Текстови променливи

Ако дадено съдържание се повтаря многократно в текста на записката, може да го присвоите на текстова променлива и да го преизползвате многократно. Освен, че си спестявате писане или копиране, размерът на файла става по-малък, а ако се наложи в бъдеще да промените нещо, ще го направите само на едно място, а не и целия текст на записката. Дефинирането на текстови променливи, става по следния начин:

Текстова променлива от един ред:

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

Текстова променлива на няколко реда:

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

На този етап, съдържанието може да бъде всякакъв текст: част от коментар, код или и двете. За да го използвате, въведете името на променливата на съответното място по-нататък в записката. Преди компилирането и, Calcpad ще замести съответното съдържание на всички места, където се среща името на променливата. След заместването, трябва да се получи валиден код на Calcpad.

Пример 1: Резултат от оразмерителна проверка:

```
#def CHK_OK$ = 'Проверката е удовлетворена
#def CHK_NOT_OK$ = 'Проверката не е удовлетворена
σ = 100MPa','σ_u = 160MPa
#if σ ≤ σ_u
    CHK_OK$
#else
    CHK_NOT_OK$
#end if
```

Макроси

Макросите се различават от текстовите променливи по това, че имат и параметри. Те се посочват в скоби след името на макроса и могат да участват в съдържанието му. Съответно, при извикване на макроса, трябва да бъдат въведени и стойностите на параметрите. Дефинирането на макрос става по един от следните начини:

Макрос на един ред:

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

Макрос на много редове:

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

Имената на текстовите променливи, макросите и техните параметри могат да включват само главни и малки букви на латиница и долна черта „_”. Те задължително трябва да завършват със символа „$“. Може да видите кода, който се генерира след заместването на текстовите и макросите, като включите отметката ☑ **Разгънат код** и стартирате изчисленията. Отметката се показва автоматично под прозореца с резултати, винаги, когато в кода има макроси.

Пример 2: Макрос за проверка на напрежения

```
#def CHECK$(x$; x_u$)
    #if x$ ≤ x_u$
        CHK_OK$':'x$'&le; 'x_u$
    #else
        CHK_NOT_OK$':'x$'&gt; 'x_u$
    #end if
#end def
τ = 80MPa','τ_u = 120MPa
CHECK$(σ; σ_u)
CHECK$(τ; τ_u)
```

Може да запишем като макрос целия условен блок за проверка. Оттук-нататък, за да проверим напреженията, ще ни трябва само по един ред код със съответните напрежения като параметри, както е показано на редове 17 и 18. Тук използваме и дефинираните в

предишния пример текстови променливи. Резултатът от изпълнението на редове 16 и 17 ще бъде генерирането на следния разгънат код:

```
 9  #if σ ≤ σ_u
10      'Проверката е удовлетворена'':'σ'&le; 'σ_u
11  #else
12      'Проверката не е удовлетворена'':'σ'&gt; 'σ_u
13  #end if
14  #if τ ≤ τ_u
15      'Проверката е удовлетворена'':'τ'&le; 'τ_u
16  #else
17      'Проверката не е удовлетворена'':'τ'&gt; 'τ_u
18  #end if
```

Макросите и текстовите променливи могат да използват вече дефинирани такива в съдържанието си, но не и да включват дефиниции на други макроси и текстови променливи. Те трябва да са дефинирани извън „тялото“ на макроса. Те могат да включват и референции към модули с `#include`, но само ако външния модул не съдържа макроси.

Модулите и макросите се обработват и заместват на предварителен етап, след което генерираният „разгънат“ код на записката се изпълнява както обикновено.

Друго приложение на макросите е да си направите кратки „помощници“ за Html форматиране, например:

Пример 3: Html помощници

```
1  #def b$(x$) = <strong>x$</strong>
2  #def i$(x$) = <em>x$</em>
3  #def sup$(x$) = <sup>x$</sup>
4  #def sub$(x$) = <sub>x$</sub>
5  #def err$(x$) = <span class="err">x$</span>
6  '<p>b$(Проверка): i$(f)sub$(yk) = err$(2) kN/cmsup$(2)</p>
```

Разгънатият код, генериран от ред 6, е както следва:

```
'<p><strong>Проверка</strong>: <em>f</em><sub>yk</sub> = <span
class="err">2</span> kN/cm<sup>2</sup></p>
```

Той е доста по-дълъг от изходния, което значи, че ни спестява и доста писане. Съответно, крайният резултат от неговото изпълнение ще бъде:

Проверка: $f_{yk} = 2$ kN/cm^2

Така, на практика може да съставите свой собствен макро език, да го запишете във вид на модули и да го ползвате във всяка записка, като добавите референция с `#include` най-отгоре. Трябва да се има и в предвид, че прекаленото използване на макроси прави съдържанието по-трудно за четене, проверка и откриване на грешки. За тази цел, е необходимо да подбираме внимателно имената на макросите и параметрите, така че да са разбираеми.

Производителност на ядрото

В последните няколко версии беше извършена значителна работа по подобряването на производителността на изчислителното ядро. Като резултат, скоростта на работа е увеличена около два и повече пъти, спрямо версия 5.7. Това се отнася най-вече за случаите, когато се използват изчисления с мерни единици и числени методи (виж следващата точка).

Като цяло, Calcpad може да изчислява сума или произведение 100 – 300 млн. пъти в секунда, на стандартен лаптоп (2022 , i5 или i7). Събирането и изваждането са по-бързи от умножението и делението. Ако има мерни единици, скоростта се забавя с около 50% при събиране и изваждане, до няколко пъти при умножение и деление. За повечето задачи от строителното проектиране обаче, изчисленията приключват почти мигновено.

Числено интегриране

Добавена е нова команда за числено интегриране, като алтернатива на `$Area`. Тя има следния вид:

```
$Integral{f(x) @ x = a : b}
```

Командата `$Area` използва адаптивната квадратура на Гаус-Лобато-Кронрод (Gander & Gautschi, 2000). Тя като цяло е доста ефективна, дори и за прекъснати функции.

Новата команда `$Integral` използва Tanh-Sinh квадратура (Takahashi & Mori, 1974), която използва двойно-експоненциална трансформация. Отчетени са и последващите подобрения от Michashki & Mosig (2016) и Van Engelen (2022). Процедурата в Calcpad е допълнително оптимизирана, като абсцисите и теглата се изчисляват предварително и се кешират. Този алгоритъм превъзхожда значително командата `$Area`, но е приложим само за **непрекъснати** и **гладки** функции. Ако функцията не удовлетворява горните условия, този метод **не трябва** да се използва.

Форматиране на интеграли, суми и произведения

Добавено е фигурно форматиране на интегралите, сумите и произведенията, със символ и граници под и над символа, както следва:

$$\int_a^b f(x)\,\mathrm{d}x = 2 \qquad \sum_{k=1}^{n} f(k) = 6 \qquad \prod_{k=1}^{n} f(k) = 6$$

Оператори

Операторите за сравнение в Calcpad използват следните символи: ≡ ≠ ≤ ≥. Те са по-кратки и прегледни за включване в записката. Проблемът е, че тези символи ги няма на клавиатурата, и трябва да се избират с бутон. Като алтернатива, може да използвате и аналогичните оператори от C/C++, съответно, както следва: == != <= >=.

Мерни единици

Единици за ъгли

В Calcpad може да задавате мерните единици за ъгли по три различни начина:

1. Чрез радио бутоните от прозореца на приложението: ◉ **D**, ◉ **R**, ◉ **G**. Това е препоръчително само когато използваме Calcpad за прости изчисления като обикновен калкулатор.
2. Чрез команди в текста на записката: `#deg` - градуси, `#rad` - радиани, `#gra` – градуси. Този начин трябва да се прилага при разработване на изчислителни записки, особено когато са предназначени за други потребители. Мерните единици в текста на записката са меродавни пред настройките от радио бутоните.
3. Чрез стандартни мерни единици. Те се добавят непосредствено след стойността на ъгъла. Тази опция е удобна, когато имаме голямо разнообразие от ъгли в различни мерни единици, искаме да се виждат в записката или се налага често да конвертираме между тях. Това може да става автоматично, както и при останалите. В последната версия може да ползвате следните мерни единици:

   °, *deg* – градуси;

   ′ – минути = 1/60°;

   ″ – секунди = 1/60′;

   rad – радиани;

   grad – градуси;

   rev – обороти.

Мерните единици, зададени непосредствено след стойностите, са с приоритет спрямо предишните две опции. Обратните тригонометрични функции могат да връщат бездименсионни стойности, или стойности с мерните единици по подразбиране, зададени чрез `#deg`, `#rad`, `#gra` или радио бутоните. По подразбиране, връщаните стойности са бездименсионни. Ако искаме да връщат стойностите с мерни единици, трябва да дефинираме променливата: `ReturnAngleUnits = 1`.

Единици за сила

В последната версия е добавена и единицата *gf* – грам-сила;

Единици за електричество

Добавени са нови мерни единици и техните производни:

- пълна мощност (Волт-Ампер):
  VA, *kVA*, *MVA*, *GVA*, *TVA*, *mVA*, *μVA*, *nVA*, *pVA*;
- реактивна мощност (Волт-Ампер реакт.):
  VAR, *kVAR*, *MVAR*, *GVAR*, *TVAR*, *mVAR*, *μVAR*, *nVAR*, *pVAR*;
- проводимост – алтернативно на Сименс е добавена и старата единица (мо):
  ℧, *k*℧, *M*℧, *G*℧, *T*℧, *m*℧, *μ*℧, *n*℧, *p*℧;

Оправен е и проблема с изходните мерни единици от формулите. В по-старите версии, резултатът от изчисленията се декомпозираше до комбинация от базовите физични дименсии, например, вместо W - $kg{\cdot}m^2{\cdot}s^3$. Сега вече се извежда в съответните композитни единици, в които е получен: W, V, Ω, S и пр.

Html форми и UI

С Calcpad може да създадете Html форма за вход на данни за всяка записка. Освен падащи списъци и полета за въвеждане, в новата версия лесно може да се добавят отметки ☑ и радио бутони ◉.

Аналогично на падащите списъци, отметките и радио бутоните също трябва да са привързани с поле за въвеждане, където да се записва стойността. То трябва да е поставено в обвиващ Html елемент който да има **id**, като същото **id** трябва да се въведе като **name** или **data-target** на отметката/радио бутона. Съответно, отметката/радио бутонът трябва да има и зададена стойност **value**, която при кликване да се попълни автоматично в полето. За добавянето на отметки и радио бутони може да използвате следния примерен код:

Пример 4: Радио бутони: Клас на стоманата: ◉ S235, ○ S235, ○ S235

```
'<p>Клас на стоманата:
'<input name="steel" type="radio" id="S235" value="235" >
'<label for="S235">S235</label>
'<input name="steel" type="radio" id="S275" value="275" >
'<label for="S275">S275</label>
'<input name="steel" type="radio" id="S355" value="355" >
'<label for="S355">S355</label></p>
'<p id="steel" style="display:none;">'f_y = ? '</p>
```

Пример 5: Отметки: Опции за профила: ☑ Горещо валцуван

```
'Опции за профила:
'<p><input name="Rolled" type="checkbox" id="roll" value="1" >
'<label for="roll">Горещо валцуван</label></p>
'<p id="Rolled" style="display:none;">'Rolled = ? '</p>
```

Падащите списъци, отметките и радио бутоните се заключват след изчисляването на записката, като направеният избор се запазва. Ако към падащия списък добавим свойството `class="post"`, той се превръща в текст. Така отпада необходимостта тези елементи да се слагат в секция `#pre`, а в пост да се добавят текстовете в `#post`, както досега.

Външни препратки (линкове)

В новата версия е оправен и проблема с отварянето на препратки (линкове) към външни сайтове, например справки за таблици с материали, натоварвания, коефициенти и др. Може да избирате и с кой браузър да се отварят линковете от падащия списък под прозореца с резултати. Той включва шест от най-популярните уеб браузъри, както следва: Chrome, MS Edge, Firefox, Safari, Opera и Internet Explorer (IExplore).