Интерактивно програмиране в система Mathematica

I. Кратко въведение в система Mathematica

1. Аритметика: Синтаксис, видове данни, оператори и изрази

В тази част ще покажем как Mathematica може да се използва като графичен калкулатор за обработка и визуализиране на различни изрази.

Интерфейс (front end): Работа с програмата, запазване и принтиране

Поздравления! Току що отворихте първия си документ (notebook) в Mathematica.

Преди да започнем ще обърнем внимание на някои основни неща при използването на Mathematica. Терминът front end се отнася до потребителския интерфейс на програмата. В нашия случай "front end" на Mathematica е оформлението на прозорците, лентата с менютата и начина по който въвеждаме входните данни и получаваме обработения резултат чрез операционната система.

Работа със система Mathematica

Стартирайте система *Mathematica*. Това може да направите от менюто “Старт” или от икона за бърз достъп (shortcut). Работните документи на програмата се наричат notebook (тетрадка или бележник). Пред Вас ще се отвори празен документ подобен на белия лист за писане. Всеки документ се състои от двойки входни и изходни клетки. Клетка е мястото в което може да въвеждаме команди или да получаваме резултата от обработката им. Ще забележите, че клетките имат номера “In[1]” и “Out[1]”.
Работните клетки могат да бъдат използвани за получаване на повече информация относно стойността на определена константа или действието на определена команда. Това става чрез поставянето на въпросителен знак пред константата, променливата или командата. Например:

In[1]:=
```
? Pi
```

Pi is π, with numerical value ≃ 3.14159. ≫

Повече информация се получава при поставянето на втори въпросителен знак.

In[2]:=
```
?? Plot
```

Plot[f, {x, x_{min}, x_{max}}] generates a plot of f as a function of x from x_{min} to x_{max}.
Plot[{f_1, f_2, …}, {x, x_{min}, x_{max}}] plots several functions f_i.
Plot[…, {x} ∈ reg] takes the variable x to be in the geometric region reg. ≫

```
Attributes[Plot] = {Protected, ReadProtected}
```

По този начин ще получите повече информация относно командата Plot заедно със синтаксиса. Подробна информация и помощ може да получите от менюто Help. За изпълнението на някои команди от съществено значение е версията на програмата, с която разполагате. Повече информация може да получите при изпълнението на командата $Version.

```
In[3]:= $Version
Out[3]= 10.1.0  for Microsoft Windows (64-bit) (March 24, 2015)
```

Mathematica разпознава основните оператори +, -, *, /, и ^ за степен. Също така и основните константи като например: π (въвежда се като Pi), експонента e (въвежда се като E или Exp), безкрайност (въвежда се като Infinity), имагинерната единица i (въвежда се като I). Системата разпознава и огромен брой от математически функции като например: Sin[x], Cos[x], ArcSin[x], Log[x], Factorial[n] (или по-просто записано като n!) и много много други. *Mathematica* знае стойностите на тези функции за определен набор от променливи. Обърнете внимание, че всяка команда започва с главна буква, а аргументите са в квадратни скоби. Вида на използваните скоби е от съществено значение тъй като обозначават различни неща неща за системата. Обикновените скоби () се използват за групиране и указване на приоритет на действията.

Използването на знакът равно (=) след променливата x, например, означава, че задаваме постоянна стойност на променливата.

```
In[4]:= x = 7
Out[4]= 7
```

Тази стойност на променливата *Mathematica* ще помни до затварянето на програмата и можем да я използваме за всички последващи операции с използването на x.

```
In[5]:= x + 3
Out[5]= 10
```

По същия начин можете да задавате постоянни стойности и на други променливи в рамките на една сесия.

```
In[6]:= y = 20
Out[6]= 20

In[7]:= x * y
Out[7]= 140

In[8]:= (x + 3) * y
Out[8]= 200

In[9]:= 2 ^ x
Out[9]= 128
```

В някои случаи задаването на постоянна стойност на аргумента може да доведе до грешки при изпълнението на определени команди. Например интегриране:

```
In[10]:= Intergrate[x^2, x]
Out[10]= Intergrate[49, 7]
```

Вместо очаквания резултат получаваме друг, който очевидно е грешен.
Когато повече не Ви е необходима въведената стойност за x може да се освободите от нея чрез изпълнението на командата Clear [x] или Clear[y] и т.н. Ако искате системата да забрави всички стойности въведени по време на сесията командата придобива вида Clear[“Global’*”]. Понякога

се препоръчва изпълнението на тази команда в началото на всеки notebook с цел предотвратяване на последващи конфликти.

```
In[11]:= Clear["Global'*"]
```

След изпълнението на командата Clear ако се върнете в клетката и обработите отново командата за интегриране ще получите правилния резултат.

В началото обърнахме внимание, че работните клетки си имат име и номер в квадратни скоби. Този номер може да използвате за да направите препратка към съответната клетка с цел резултата от обработката на тази клетка да бъде използван при операция в друга клетка. Например:

```
In[12]:= 5
Out[12]= 5

In[13]:= % + 15
Out[13]= 20

In[14]:= % * 100
Out[14]= 2000

In[15]:= %% * 2
Out[15]= 40
```

Използването на % без номер на клетка означава системата да използва стойността в предишната клетка, %% - стойността в по-предишната клетка. Ако е неоходимо да се направи адресиране към точно определена клетка след % се изпизва номера на клетката. Например %7 ще използва резултата от клетка Out[7].

Клетки и обработка

Основният принцип, по който се разделят работните документи в система Mathematica е разделянето им на клетки. Една клетка е мястото, в което можем да въвеждаме команди на система *Mathematica* или получаваме резултата от обработката. Ще забележите, че в дясната част на клетката се появява скоба, която обхваща цялото съдържание на клетката. Тази скоба определя обхвата на клетката, в която сте въвели информация. Обърнете внимание, че вътрешните скоби на тази клетка (от дясно на екрана) има двойна (допълнителна) чертичка. Тази допълнителна чертичка показва, че е въведено допълнително ограничение. В този случай *Mathematica* е принудена да игнорира съдържанието на клетката като инструкции или команди. Това е просто “текстова клетка”. Клетките, които искаме да обработим с *Mathematica* не трябва да бъдат текстови клетки, а входни клетки. По подразбиране клетките, които съдържат информация за обработка се визуализират в синьо (въпреки че това може да бъде променяно от настройките на програмата). Преди да се научим да създаме свои клетки за обработка ще обработим някои вградени в този документ клетки.

За да обработите клетката кликнете с мишката вътре в клетката, след което натиснете Enter от цифровата клавиатура (или Shift + Enter от основната клавиатура). Уверете се, че курсурът на мишката се намира в клетката, която искате да обработите. В противен случай нищо няма да се

случи. Същото действие може да бъде извършено и от менюто Evaluation от лентата с менюта.

Нека да видим как става това като обработим клетката по-долу.

```
In[16]:= 4*5^2+13
Out[16]= 113
```

Ще Ви направи впечатление, че отнема доста дълго време на системата да обработи тази проста операция. Не се тревожете! Това е така, защото Mathematica натоварва компютъра Ви на два етапа. Програмата се състои от две части: потребителски интерфейс (front-end) и ядро (kernel). При първоначалното стартиране на системата се зарежда само потребителския интерфейс. Частта от системата, която всъщност прави изчисленията не се зарежда докато за първи път не натиснете Shift + Enter. Забележете, че входните и изходните клетки са с надпис In[1] Out[1]. Всяка клетка е заградена от скоба, а двете клетки са обхванати от обща скоба, т. е. документа има йерархична структура.

Аритметични операции

В тази част ще покажем как *Mathematica* може да се използва като обикновен научен калкулатор, изчислявайки изрази. Ще бъдат показани операторите, които обикновено са символни, като например "+" , който показва на *Mathematica*, че трябва да извърши определен вид задача или операция (в този случай събиране).

Аритметичните оператори имат стандартна за калкулаторите форма ("+", "-", "*", "/" и "^") и се спазва стандартно математическо предимство при извършване на операциите, което означава например, че умножението и делението се изпълняват преди събирането и изваждането. Експонентата се изпълнява преди умножението и делението, които имат същото ниво на приоритет. При изразът "4*5^2+13" трябва първо да се изпълни "5^2" след това "4*(5^2)" и накрая да се добави "13".

```
In[17]:= 4*5^2+13
Out[17]= 113
```

Mathematica приема и някои нестандартни форми при аритметични операции. Така например знакът за умножение * може да бъде пропуснат и въпреки това да се извърши операция умножение. Необходимо е да оставите разстояние на негово място за да може *Mathematica* да разбере, че искате да направи умножение, не че изписвате просто числото 45. Обработете следващата клетка:

```
In[18]:= 4  5^2+13
Out[18]= 113

In[19]:= 4 × 5^2 + 13
Out[19]= 113
```

Забелязвате, че при оставяне на разтояние системата автоматично го заменя със знака за умножение. При умножението на променлива с число разстоянието може да бъде пропуснато, като числото трябва да бъде изписано преди променливата.

```
In[20]:= 3x + 5x
Out[20]= 56
```

В противен случай "x3" и "x5" ще бъдат интерпретирани като две различни променливи (в този случай 3 и 5 ще бъдат разпознати като долни индекси) и по този начин те не се комбинират. Докато обработката на клетката по-горе дава резултат "8x", изпълнението на клетката по-долу просто Ви връща резултат "x3+x5".

```
In[21]:= x3 + x5
Out[21]= x3 + x5
```

Булеви оператори

Mathematica също има и някои функции, които няма да Ви върнат числена стойност, а вместо това ще получите резултат “вярно” или “невярно” (“True” и ”False”). Тъй като “вярно” и “невярно” се считат за булеви стойности, тези функции се наричат също булеви. Вие вече познавате повечето от тях: по-малко, по-голямо и т.н. Релационните оператори сравняват два аритметични израза и включват: равен на "==" (изписва се като два знака равно, и се използва да “попитаме” дали два израза са равни), неравенство ("!="), по-голямо (">"), по-малко ("<"), по-голямо или равно (">=") и по-малко или равно ("<="). Логическите оператори, от друга страна, манипулират стойностите “вярно” и “невярно”, като отново връщат резултат “вярно” или “невярно”. Наборът от логически оператори в *Mathematica* включва операторите “и” ("&&"), който дава резултата “вярно” само когато и двете стойности са “вярно”, “или” ("||"), който дава резултата “вярно”, когато поне едната от двете стойности (или и двете заедно) са “вярно”. Нека покажем действието на тези булеви оператори:

```
In[22]:= 3 > 5
         5 > 5
         5 >= 5
         5 == 5
         5 != 5
         ( 3 > 6 ) || ( 4 < 5 )
Out[22]= False
Out[23]= False
Out[24]= True
Out[25]= True
Out[26]= False
Out[27]= True
```

Обърнете внимание на някои особености! При проверката за равенство знакът за равно е двоен, удивителния се използва за отрицание. Операторите могат да бъдат комбинирани в рамките на един изразв както е показано в примерите по-долу.

```
In[28]:= ( 2 > 1 ) && ( 3 > 4 )
Out[28]= False
```

В този пример първо се обработват операторите в скобите, които връщат резултат “вярно” и “невярно”, а след това логическия оператор “и” дава краен резултата “невярно”, защото и двете стойности не са “вярно”.

```
In[29]:= ( 2 > 1 ) || ( 3 > 4 )
Out[29]= True
```

По същия начин, в горния пример, резултатът от изразите в скобите е “вярно” и “невярно”, съответно, след което оператора “или” ни връща резултат “вярно” защото поне една от булевите стойности е “вярно”.
Какво ще се случи ако *Mathematica* не може да прецени дали една стойност е по-голяма от друга? В този случай тя просто оставя израза необработен. Опитайте!

```
In[30]:= Pi < 22 / 7
         x < 22 / 7
Out[30]= True
Out[31]= False
```

Предимство на операторите

Вече демонстрирахме предимството при използването на аритметичните операции. Но как Mathematica решава коя операция да обработи преди другите операции? Mathematica има дефинирани правила, които определят относителното предимство на всички нейни оператори, което означава, че предварително знае кои операции трябва да обработи първо при обработката на израз.

Аритметическите оператори имат предимство пред релационните (операторите за сравняване) и логическите.

```
In[32]:= 3 > 2 + 2
Out[32]= False
```

Изразът по-горе дава резултат “невярно”, имайки предвид, че първо се обработва събирането на “2+2”, което е равно на 4 и след това се прави проверката дали 3 е по-голямо от 4.

След това, в рамките на булевите оператори, операторите за сравнение имат предимство пред логическите.

```
In[33]:= 5 > 4 && 1 > 2
Out[33]= False
```

В този случай всеки от изразите “по-голям” е обработен първо, така израза добива вида "True && False"; след това се изпълнява оператора "и" и се получава резултат "невярно".

Точни vs. приблизителни стойности

В *Mathematica* има два вида стойности: точни и приблизителни. Точните стойности обикновето заемат повече място за съхранение в оперативната памет, а изчисленията с точни стойности отнемат значително повече време, за да се осигури и точен резултат. Приблизителните

стойности пък обратно - заемат по-малко оперативна памет и отнемат по-малко време за обработка тъй като приблизителния резултат е този, който е необходим (а в определени случаи и възможен). Това е първият компромис, който трябва да направим при програмирането. Трябва да решим дали ни е необходим по-точен резултат или по-малко време за обработка.

При изпълнението на всички операции системата се старае да запази възможно най-голяма точност, освен ако не поискате друго от нея. Това може да доведе до известно объркване, че системата не може да пресметне исканата от Вас стойност.

Ако изразът съдържа само точни стойности, *Mathematica* никога няма да направи приближение и ще обработи израза до възможно най-простата точна форма.

```
In[34]:= 2^100/6
Out[34]= 633 825 300 114 114 700 748 351 602 688
                                    3
```

Ако изразът съдържа поне една приближена стойност то и резултатът ще бъде изведен в приближена стойност. Ако аргументът на командата се зададе с десетично число, само с поставянето на десетична точка, системата разпознава въведените данни като приближени и извежда резултат също в приближена стойност.

```
In[35]:= 2^100/6.
Out[35]= 2.11275 × 10^29
```

Но *Mathematica* може да прави много повече от просто събиране и умножение на числа. Опитайте:

```
In[36]:= 102/9
Out[36]= 34/3
```

Забелязвате, че *Mathematica* прави опростяване на дробта до най-проста форма (т.е. до най-малката стойност на числителя и знаменателя). Също така забелязвате, че резултатът все още се извежда като дроб, а не като десетично число. Това е така, защото това е най-точната стойност, която може да бъде получена.

Ако искаме да преобразуваме точен или частично точен израз в призлизителна стойност, използваме функцията N[] . Обърнете внимание, че всяка команда започва с главна буква, а аргументите на командата (функцията) са в квадратни скоби. Обикновените скоби се използват за указване на приоритет на действията. По подразбиране програмата работи с точност пет знака след десетичната запетая, но това може да бъде променяно по всяко време.

```
In[37]:= 2^100/(6. Pi)
Out[37]= 6.7251 × 10^28

In[38]:= N[2^100/(6. Pi)]
Out[38]= 6.7251 × 10^28
```

Много е важно да се разбере разликата между приближена и точна стойност. Приближенията, които веднъж се наложат в израза, принуждават програмата да обработи този израз до определен брой значещи цифри след десетичната запетая. След като бъде изчислена и представена приблизителната стойност на израза, не може да бъде повишена точността при

следващи обработки. Само точните стойности запазват своята “безкрайно голяма” точност при всички изчисления.

Можем да покажем разликата между точни и приблизителни стойности като използваме функцията "N[]". Не забравяйте, че функцията "N[]" взима точната стойност на израза и се опитва да намери приближение с желания брой цифри. След израза се поставя запетая и след това броя на цифрите. Функцията "N[]" работи най-добре ако израза е точен, така може да намерим приближената стойност с толкова значещи цифри, колкото бихме желали. В някои случаи обаче това може да създаде проблеми тъй като е възможно да не съществува приближена стойност с повече значещи цифри.

Нека разгледаме два случая. Първо ще поискаме стойността на "2^100/(6*Pi)" с точност 50 знака след десетичната запетая.

In[39]:=
```
N[2^100/(6 Pi),50]
```

Out[39]= $6.7250953046576938310577364930610450566081011366130 \times 10^{28}$

Резултатът е точно това, което искахме. А сега ще поискаме резултата първо с точност 16 знака, а след това с точност 50 знака. Вижда се, че веднъж въведено ограничение на точността, след това по-голяма точност не може да бъде получена.

In[40]:=
```
N[N[2^100/(6 Pi),16],50]
```

Out[40]= $6.725095304657694 \times 10^{28}$