Запознаване
с
математическия език
на
Wolfram *Mathematica*.

Основни предимства на Mathematica

Директна комуникация с ядрото:

Командите се въвеждат от клавиатурата в бележника и директното се изпълняват от ядрото

Елементи на командите

1/ ***Числа*** (цели 2, рационални ½ , реални 2.3, комплексни1+3i)

2/ ***Променливи*** (a, x, var1)

3***/ Специални константи*** (Pi, E)

4/ ***Аритметични операции*** (+, -, *, /, ^)

5/ ***Операции за отношение*** (<,<=,>,>=)

6***/ Логически операции***(&&, ||, !)

7***/ Низове*** (“test”)

8***/ Специални символи и скоби*** (:, ;, %)

9/ ***Имена на вградени и потребителски функции*** (Sin, Plot)

Запознаване с елементите на командите

1/ ***Числа*** : цели 2, рационални ½ , реални 2.3, комплексни1+3i

2/ ***Променливи*** (a, x, var1)

Да припомним:

- *Mathematica различава главни и малки букви: sin е различно от Sin.*
- Имената на променливите започват ЗАДЪЛЖИТЕЛНО с буква (малка или голяма), последвана от неограничен брой букви и/или цифри. НИКОГА НЕ започва с цифра

3/ ***Специални константи :*** три важни константи π, e, *i*

Въвеждане от клавиатурата: Pi, E, I (важно е да са главни букви)

```
In[14]:= π
Out[14]= π
```

```
In[15]:= π + 0.
Out[15]= 3.14159
```

```
In[18]:= Pi + 0.
Out[18]= 3.14159
```

```
In[20]:= E + 0.
Out[20]= 2.71828
```

```
In[22]:= I + 0.
Out[22]= 0. + 1. i
```

Ако искаме да получим повече знаци след десетичната запетая, използваме N

```
In[26]:= N[Pi, 10]
Out[26]= 3.141592654
```

```
In[27]:= N[Pi, 100]
Out[27]= 3.14159265358979323846264338327950288419716939937510582097494459230781640628620
      8998628034825342117068
```

4/ ***Аритметични операции*** (+, -, *, /, ^)

Mathematica работи като калкулатор като използва основните аритметични действия ***+, –, *, /***

Точката се използва вместо десетична запетая.

<u>Някои примери (с особености):</u>

In[3]:= `1 / 16`

Out[3]= $\frac{1}{16}$

In[4]:= `2 / 16`

Out[4]= $\frac{1}{8}$

In[5]:= `2. / 16`

Out[5]= `0.125`

Това изглежда странно. Но *Mathematica* разбира десетичните числа като приближения и обикновено ги избягва в изхода, т.е. ако изхода е цяло число, може да бъдете сигурни, че това е точната стойност. Дробите се представят като ***несъкратими*** дроби.

Разширяване на клавиатурата чрез използване на палетите

In[6]:= $\sqrt[3]{32}$

Out[6]= $2 \times 2^{2/3}$

In[7]:= $\sqrt[3]{32.}$

Out[7]= 3.1748

In[8]:= $\mathbf{N}\left[\sqrt[3]{32}\right]$

Out[8]= 3.1748

Направи разликата между трите входа- за да получим приближения числен резултат използваме или десетичната точка или вградената функция N

<u>ВАЖНО:</u> Използването на десетичната ТОЧКА дава възможност да получим числената стойност на израза.

5/ ***Операции за отношение*** **(==, <, <=, >, >=, !=)**

Използват се за сравняване на числа
и са от вида

x==y	равно
x!=y	различно (може и като x≠ y)
x>=y	по-голямо или равно и

т.н.

x==y==z	всички са равни
x!=y!=z	всички са различни
x<y<z	строго растяща редица

Резултатът е или TRUE (верен) или FALSE (неверен).

Примери за операции за отношение

```
In[16]:= 10 < 7
Out[16]= False
```

Числото 10 не е по малко от числото 7 и затова резултата е False

```
In[17]:= 8 < ∛1025
Out[17]= True
```

Първото число е по малко от второто и затова резултата е True

```
In[18]:= 3 != 2 != 3 !
Out[18]= True
```

Всичките числа не са равни и затова резултата е True

```
In[19]:= 25 < 30 <= 43
Out[19]= True
```

Могат да се смесват операциите <, > с <= и с >=

```
In[20]:= Pi^E < E^Pi
Out[20]= True
```

Могат да се сравняват е специалните константи

```
In[21]:= a < b
Out[21]= a < b
```

Не се знае стойността на променливата a и на променливата b

```
In[23]:= a = 2.4; b = 3.5; a < b
Out[23]= True
```

НО, ако се присвоят предварително стойности на двете променливи, тогава те могат да се сравняват

6/ ***Логически операции (Булеви операции)*** (&&, ||, !)

Логическите операции са

!p	отрицание (НЕ)(не p)	
p||q	И	(верно, когато поне един от аргументите е верен)
p&&q	ИЛИ	(верно, когато И двата аргумента са верни)

където *p и q са операции за отношение*

Резултатът винаги е или *True или False*

Например,

ако искаме да запишем, че стойностите на променливата x са в интервала (1,3)то записваме

X>1§§X<3

ако искаме да запишем, че стойностите на променливата x са в интервала (1,3) или в интервала (5,6) пишем

(X>1 §§ X<3)||(X>5 && X<6)

Примери за логически операции

```
In[29]:= ! 2 < 3
Out[29]= False

In[30]:= 2 < 3 || 3 < 5
Out[30]= True

In[31]:= 2 < 3 && 3 ≠ 2
Out[31]= True

In[32]:= p && q
Out[32]= p && q

In[33]:= p = 2; q = 3; p && q
Out[33]= 2 && 3

In[34]:= p = 2 < 3; q = 3 < 4; p && q
Out[34]= True

In[35]:= r = 4 ≠ 5; p || q && r
Out[35]= True
```

Отрицанието на отношението < е >== което в случая не е верно

Не са известни p и q и затова не се извършва операцията, а само се копира както е записана

p и q са числа а не операции за отношение и затова не се извършва логическата операция

В този случай p и q операции за отношение и затова се извършва операцията и се дава резултата

Може да се съчетават по няколко логически операции

7/ ***Низове/стрингове*** (“test”) Текст заграден в кавички

Може да съдържа всяка редица от обикновени или специални символи , заградени в кавички

ПРИМЕРИ:

Когато въвеждате стринг винаги трябва да го заграждате в кавички. Докато при извеждането на стринг, *Mathematica може и да не съдържа кавички.*

```
In[1]:= “Това е стринг."

Out[1]= Това е стринг.
```

Но вие може да накарате *Mathematica* да ви покаже кавичките чрез използване на InputForm [spring]

```
In[2]:= InputForm[%]
Out[2]//InputForm=
“Това е стринг."
```

7/ ***Низове/стрингове*** (“test”)- продължение

Това, че *Mathematica* не показва винаги кавичките позволява стринговете да се използват като текст.

```
In[3]:= Print[“Стойността е", 567, "."]
                Стойността е 567.
```

ВАЖНО. Независимо, че понякога в изхода стринга не е заграден от кавички, той си е различен от променливите.
Например, стрингът "x" често в изхода се записва като x, но е различен от променливата x.

```
In[4]:= "x" ==x
Out[4]=  False
```

Ако искате да надпишете в изход-клетката резултат, то може да използвате string и вградената функция **StringForm**

```
In[3]:= StringForm[“Стойностите са x=`2` and y=`1`.",10,5]

Out[3]= x = 5,    y =  10
```

Внимание: номерът на израза е заграден със символа **`, а не с ‘**

8/ ***Специални символи и скоби*** (:, ;, %)

Да припомним за използването на скоби

Кръгли(малки) скоби (x) се използват за промяна на реда на действията в математически изрази; sin(*x) ще се интерпретира като* $\sin * x$

Квадратни скоби [x] се използват за отделяне елементите на функцията, името й от аргументите й, например Sin[x]

Двойни квадратни скоби [[x]] се използват за индексиране, например при посочване елемент на матрица

Големи(криви) скоби{x} се използват при конструирането на списъци, например: {a, b, c, {d, e}}

In[10]:= 3 + 1 / 2

Out[10]= $\frac{7}{2}$

In[12]:= (3 + 1) / 2

Out[12]= 2

In[13]:= $\frac{3+1}{2}$

Out[13]= 2

Този пример показва предимството на използването на палетите

Примери за използване на скоби

```
In[1]:= (2 - 6) ^ 3 + 8
Out[1]= -56
```

Смяна реда на действията в математически израз

```
In[3]:= u = {10, 11, 12, 13}
Out[3]= {10, 11, 12, 13}
```

Дефиниране на списък

```
In[4]:= u[[2]]
Out[4]= 11
```

Индексиране на списък

```
In[5]:= Plus[2, 3]
Out[5]= 5
```

Извикване на вградена функция

```
(*Komentar*)
```

Коментар

Специални символи : ; и %

Символът % *служи за използване на резултат от предишен изход.*

% - за използване на резултата от изхода с предходен номер; може да се използва и Out[x] вместо него

%% - за използване резултата на изхода с по-предходен номер

%x - служи за използване на резултат на Out[x]; може директно да се използва и Out[x]

Символът : *служи за присвояване*

Символът := *служи за «отложено присвояване» (при дефиниране на потребителска функция)*

Символът ; *служи за забрана извеждането на резултат на екрана. Служи и за разделител на два израза, написани на един ред*

Примери за използване на специалните символи

In[10]:= $2 / 3 + \sqrt[3]{16}$

Out[10]= $\frac{2}{3} + 2 \times 2^{1/3}$

In[11]:= `N[%]` → Дава резултата от изхода с предходен номер 10 като приближено реално число

Out[11]= 3.18651

In[12]:= `x = 5 / %` → Дели 5 с резултата от предния изход 11

Out[12]= 1.56912

In[13]:= `y = 5 / %%` → Дели 5 с резултата от по-предния изход 11

Out[13]= 1.56912

In[14]:= `v = 5 / %` → Дели 5 с резултата от предния изход 13 и го присвоява на променливата v

Out[14]= 3.18651

In[15]:= `% + %%`

Out[15]= 4.75562 → Събира резултата от предния изход 14 с резултата от по-предходния изход 13

9/ ***Имена на вградени и потребителски функции***

Имена на потребителски функции:
започват с буква (малка или голяма)

`f[x_]:=x^2` Дефиниране на функция. НЕ ЗАБРАВЯЙТЕ _ която следва след аргумента

`f[2]` Изчислява стойността на функцията при x=2, като дава 4

`f[y+z]` Работи и със символи, като замества x със израза и дава $(y+z)^2$

`f[f[f[2]]]` Допуска и работа със сложна функция, като дава 16348

`Clear[f]` Изтрива дефинираната функция (забележете не се показва изход)

Вградени функции

Имената на вградените функции ВИНАГИ започват с ГЛАВНА буква.

Някои вградени функции: Тригонометрични и логаритмична функции

Техните имена започват с главна буква и могат да се наберат директно чрез палетите.

```
In[32]:= Sin[π / 4]
```

$$\text{Out[32]=} \quad \frac{1}{\sqrt{2}}$$

```
In[43]:= N[Sin[π / 4]]
Out[43]= 0.707107
```

```
In[44]:= Sin[π / 4] // N
Out[44]= 0.707107
```

```
In[33]:= Cos[π / 12]
```

$$\text{Out[33]=} \quad \frac{1+\sqrt{3}}{2\sqrt{2}}$$

Въвеждането на градуси, (малкото кръгче) става от палета. Ако няма градус, се подразбира радиани

```
In[36]:= Log[120]
Out[36]= Log[120]
```

```
In[37]:= Log[120.]
Out[37]= 4.78749
```

Някои вградени функции за прости числа

Да припомним- просто число- което има два делителя само, т.е.
се дели само на себе си и на 1
Примери: 2, 3, 5, 7, 11, 13, 17, 19, 23, ……

Проверка дали едно число е просто

Prime Q[n], където n е числото
Резултатът е True ако n е просто число и False ако n не е просто число

PrimeQ[1234]

False

PrimeQ[12343]

True

Разлагане число на прости множители

Използва се вградената функция **FactorInteger[n]**

```
In[38]:= FactorInteger[4 832 875]
Out[38]= {{5, 3}, {23, 1}, {41, 2}}
```

Интерпретация на изхода: множителят 5 се среща 3 пъти, множителят 23 –един път и множителят 43-два пъти

Вградени функции за символно пресмятане

Разлагане на полиноми на неразложими множители

Factor[полином]

```
In[41]:= Factor[t^3 - 27]
Out[41]= (-3 + t) (9 + 3 t + t^2)
```

Представяне на полиноми в нормален вид

Expand[произведение от множители]

```
In[42]:= Expand[(-3 + t) (9 + 3 t + t^2)]
Out[42]= -27 + t^3
```

Булеви функции

Това са функции чиято стойност е True / False

Positive[x]
Negative[x]
EvenQ[x]
OddQ[x]
IntegerQ[x]
където x е число или аритметичен израз от числа

```
In[37]:= Positive[4.567]
Out[37]= True

In[38]:= Negative[4.567]
Out[38]= False

In[39]:= Negative[0]
Out[39]= False

In[40]:= EvenQ[4587]
Out[40]= False

In[41]:= EvenQ[45.88]
Out[41]= False

In[42]:= OddQ[4587]
Out[42]= True

In[43]:= OddQ[45.87]
Out[43]= False

In[44]:= IntegerQ[25.45]
Out[44]= False
```