---- INTRODUCCIÓN A MAPLE ----

Reiniciar la sesión de Maple.

Borra el contenido asignado a todas las variables y vuelve a los valores por defecto.

```
> restart:
>
```

Cargado de librerías.

Se puede cargar un paquete completo:

```
> with(combinat):
Warning, the protected name Chi has been redefined and unprotected
```

O bien sólo alguna de sus funciones:

```
> with(combinat,partition):
```

También se puede utilizar una función de un paquete de ésta forma alternativa:

```
> combinat[partition](3):
>
```

Manipulación de números.

```
> a1:=3/5:
> a2:=sqrt(2):
> a3:=(4/3)^(5/3)*3!:
> a4:=(4/3)^Pi:
> a5:=0.4523689709682790682+1/3+sqrt(2)+Pi:
>
```

Comandos numérico-simbólico.

El comando 'convert(expresión,forma)' convierte de flotante a fracción o viceversa
El comando 'evalf(expresión,nº de dígitos)' evalúa en coma flotante.

```
> a1:=21.3:
> convert(a1,fraction):
> convert(%,float):
> a2:=sqrt(5)^exp(1):
> convert(%,float):
> evalf(%%,10):
> evalf(%%%,20):
> evalf(%%%%,200):
> evalf(a2,200):
> f:=12/3*x^4+exp(1)*x^3+Pi*x+sqrt(3):
> convert(%,float):
> evalf(%%,20):
```

El comando 'convert' tiene también otras utilidades...

```
> convert(123,binary):
>
```

Comandos de manipulación de números enteros.

El comando 'ifactor(número)' factoriza números enteros.

```
> a1:=1289900: a2:=6528780: a3:=6574820:
> ifactor(a1):
> ifactor(a2):
> ifactor(a3):
```

Los comandos 'igcd(número_1,...,número_n)' e 'ilcm(número_1,...,número_n)' calculan respectivamente
el máximo común divisor y el mínimo común múltiplo de números enteros.

```
> igcd(a1,a2,a3):
> ilcm(a1,a2,a3):
> ifactor(%):
```

Dados dos números enteros 'm,n', los comandos 'iquo' e 'irem' calculan el cociente 'q' y el resto 'r' de dividir 'm' entre 'n'.
(Es decir; m=q*n+r).

```
> iquo(a3,a1):
> irem(a3,a1):
> a3-(a1*%%+%):
>
```

Comandos de definición y manipulación de funciones.

Para definir una función en Maple se pueden utilizar dos formas:

```
> f:=unapply(x^2+y^2,x,y):
> g:=(x)->sin(x):
> h:=(x,y)->exp(x)*y:
```

Maple admite las siguientes operaciones con funciones; +,-,*,/,@ (ésta última es la composición).

```
> suma:=f+h:
> suma(a,b):
> resta:=f-h:
> resta(a,b):
> producto:=f*h:
> producto(a,b):
> division:=f/h:
> division(a,b):
> composicion:=g@f:
> composicion(a,b):
>
```

Comandos sobre matrices.

Para definir una matriz en Maple hay que cargar el paquete linalg:

```
> with(linalg):
Warning, the protected names norm and trace have been redefined and
unprotected

> ?linalg,matrix:
```

Hay varias opciones para definir una matriz:
1. Escribiendo primero las dimensiones y luego todas las entradas seguidas:

```
> matrix(3,2,[1,2,3,4,5,6]):
```

2. Escribiendo cada **fila** en una lista:

```
> matrix([[1,2],[3,4],[5,6]]):
```

3. También se puede hacer esto último dando las dimensiones primero:

```
> matrix(3,2,[[1,2],[3,4],[5,6]]):
```

4. Se pueden definir las entradas como resultado de una función:

```
> f:=(i,j)->i+j:    A := matrix(3,2,f):
>
```

Comandos de manipulación de polinomios.

El comando 'randpoly([variable_1,...,variable_n])' genera un polinomio aleatorio en las variables dadas.

```
> f:=randpoly([x,y,z,t]):
```

El comando 'degree(polinomio,variable)' calcula el grado de un polinomio en una variable concreta.

```
> degree(f,x):
```

Utilizando 'degree(polinomio,{variable_1,...,variable_k})' se obtiene el grado TOTAL del polinomio en esas variables.

```
> degree(f,{x,y,z,t}):
> degree(f,{x,t}):
```

El comando 'coeff(polinomio,variable,n)' calcula el coeficiente del polinomio en el término 'variable^n'.

```
> coeff(f,x,2):
```

El comando 'coeffs(polinomio,{variable_1,...,variable_k})' calcula los coeficientes en las variables especificadas.

```
> coeffs(f,{x,y,z}):
```

Los comandos 'quo(polinomio_1,polinomio_2,variable)' y 'rem(polinomio_1,polinomio_2,variable)' calculan el cociente y el resto de la división del polinomio_1 entre el polinomio_2 respecto a la variable dada.

```
> f1:=randpoly(x):
> f2:=randpoly(x):
> quo(f1,f2,x):
> rem(f1,f2,x):
> f1-(f2*%%+%):
```

Los comandos 'gcd(polinomio_1,polinomio_2)' y 'lcm(polinomio_1,polinomio_2)' calculan el máximo común divisor y el mínimo común múltiplo de dos polinomios dados.

```
> gcd(f1,f2):
> lcm(f1,f2):
> expand(%):
```

El comando 'factor(polinomio)' factoriza el polinomio indicado sobre el cuerpo en que están sus coeficientes.

```
> factor(%):
>
```

Algunos comandos más.

El comando 'expand(expresión)' desarrolla una expresión.

```
> f:=randpoly(x):
> g:=f^3:
> expand(%):
```

Los comandos 'diff(expresión,variable)' e 'int(expresión,variable)' derivan e integran una expresión respecto de una variable.
El comando 'simplify(expresión)' simplifica una expresión.

```
> f:=sin(x)^2/(x^2+1):
> g:=diff(f,x):
> h:=int(g,x):
> f-h:
> simplify(f-h):
```

El comando 'subs({variable_1=valor_1,...,variable_n=valor_n},expresión)' sustituye en una expresión unas variables por los valores que les damos a éstas.

```
> f:=randpoly([x,y,z]):
> subs({x=2,y=3/2,x=3},f):
> subs({x=5*x+y+z,y=x+3*y-z,z=3*x+y+z},f):
>
```