


Một số lệnh cơ bản trong MAPLE

Tải Maple 12 tại <https://bit.ly/maple12> và làm theo hướng dẫn để cài đặt.

1. Để khởi tạo một phiên làm việc mới, ta chọn biểu tượng  trên thanh công cụ. Để thực hiện ta nhấn ENTER.
2. **expr**; Kết thúc **expr** bằng dấu chấm phẩy “;” để hiện kết quả của **expr**. Ví dụ **1+2;**
3. **expr**: Kết thúc **expr** bằng dấu hai chấm “:” để ẩn kết quả của **expr**. Ví dụ **1+2:**
4. **restart**; Xóa thông tin bộ nhớ.
5. **?func** Tìm hiểu về hàm **func**. Ví dụ **?sin;**. Ngoài ra, ta có thể sử dụng **Help/Maple Help** trên thanh công cụ, sau đó đánh tên hàm cần tìm hiểu vào ô tìm kiếm.
6. **var:= expr** Gán **expr** cho biến **var**. Ví dụ **x:=3+5-7;**
7. **print(expr)** In giá trị của **expr**. Ví dụ: **a:=2+5: print(a);**
8. **evalf(expr)** Tính biểu thức sang số thập phân 10 chữ số. Ví dụ: **evalf(Pi);**
9. **simplify(expr)** Rút gọn biểu thức **expr**. Ví dụ: **simplify((x+y)^2 -(x-y)^2);**
10. **expand(expr)** Khai triển biểu thức **expr**. Ví dụ: **expand((x+y)^5);**
11. **var:='var'** Giải phóng biến **var**. Ví dụ **m:=5: print(m); m:='m'; print(m);**
12. **a + b, a - b, a * b, a/b** Thực hiện phép toán cộng, trừ, nhân, chia của **a** và **b**.
13. **a ^ n** Tính a^n . Ví dụ: **4 ^ 5;**
14. Để viết một biểu thức trên nhiều dòng, ta dùng tổ hợp phím “SHIFT+ENTER” để xuống dòng

```
> 1+2-5*  
6+7;          #nhấn SHIFT+ENTER
```

15. **f(a)** Tính giá trị hàm **f** tại **a**. Ví dụ **cos(Pi);**
16. **f := x-> expr** Định nghĩa hàm một biến $f(x) = expr$. Ví dụ **bp := x-> x^2; bp(5);**
17. **f := (x, y, ...)-> expr** Định nghĩa hàm nhiều biến $f(x, y, \dots) = expr$. Ví dụ **tich := (x, y, z)-> x * y * z;**
18. **solve(eq, var)** Giải phương trình **eq** theo biến **var**. Ví dụ **solve(x^2+2*x-3=0, x);**
19. **solve(eqs, vars)** Giải hệ phương trình gồm các phương trình **eqs** = {**eq₁**, **eq₂**, ...} theo các biến **var** = {**var₁**, **var₂**, ...}. Ví dụ **solve({x^2-x+y=1, 2*x+y=3}, {x,y});**
20. **true, false** Giá trị đúng, sai. Ví dụ: **a:=true;**
21. **=, <>, <, <=, >, >=** Các phép so sánh: bằng, khác, nhỏ hơn, nhỏ hơn hoặc bằng, lớn hơn, lớn hơn hoặc bằng.
22. **seq(f(i), i = m..n)** Tạo ra một dãy số **f(i)** với **i** lấy giá trị từ **m** đến **n**. Ví dụ **seq(i^2, i = 2..10);**
23. **L:=[a, b, c, ...]** Tạo ra danh sách **L** gồm các phần tử **a, b, c, ...**. Ví dụ **L:=[2, 3, 4, 6];**
24. **seq(f(i), i in L)** Tạo ra một dãy số **f(i)** với **i** là các phần tử trong **L**. Ví dụ **seq(i^2, i in [2, 3, 5]);**

25. `nops(L)` Số phần tử của danh sách `L`. Ví dụ `L:=[2, 3, 4, 6]`: `nops(L)`;
26. `L[i]` Phần tử thứ `i` của danh sách `L`. Ví dụ: `L[3]`;
27. `for x in L do expr; od`; Thực hiện lặp đi lặp lại biểu thức `expr` với `x` lần lượt là các phần tử của `L`.

```
> L:=[3, 5, 6];
for x in L do
    print(x^2);
od;
```

28. `for i from n to m do expr; od`; Thực hiện lặp đi lặp lại biểu thức `expr` với `i` chạy từ `n` đến `m` với bước nhảy là 1.

```
> for i from -4 to 10 do
    print(i^2+1);
od;
```

29. `for i from n to m by s do expr; od`; Thực hiện lặp đi lặp lại biểu thức `expr` với `i` chạy từ `n` đến `m` với bước nhảy là `s`.

```
> for i from 3 to 10 by 2 do
    print(i^2+1);
od;
```

30. `while test do expr; od`; Nếu `test` đúng sẽ thực hiện lặp đi lặp lại `expr` cho đến khi `test` sai.

```
> n:=3;
while n<10 do
    print(n^2);
    n:=n+2;
od;
```

31. `if test then statmt fi`; Nếu `test` đúng thì thực hiện `statmt`.

```
> a:=3; b:=5;;
if a>b then
    a:=a-b;
fi;
```

32. `if test then statmt1 else statmt2 fi`; Nếu `test` đúng thì thực hiện `statmt1`, ngược lại thì thực hiện `statmt2`.

```
> a:=3; b:=5;
if a>b then
    print(a);
else
    print(b);
fi;
```

33. `func:=proc(paras) local ... expr; end proc`; Định nghĩa một hàm hay thủ tục `func` với `paras` là các tham số truyền vào.

```
> tong:=proc(a,b)
    local s;           #khai báo các biến sử dụng trong hàm
    s:=a+b;
    return s;          #trả về giá trị của hàm
end proc;
> tong(5,9);
```