BÀI 0. GIỚI THIỆU VỀ MAPLE

Maple là một phần mềm tính toán do hãng Maple Soft, một bộ phận chủ yếu của liên hợp công ty Waterloo Maple phát triển.

Cho đến nay Maple đã được phát triển qua nhiều phiên bản khác nhau và ngày càng hoàn thiện

Với phần mềm Maple, chúng ta có thể:

+ Thực hiện các tính toán với khối lượng lớn, với thời gian nhanh và độ chính xác cao.

+ Sử dụng các gói chuyên dụng của Maple để giải quyết các bài toán cụ thể như: vẽ đồ thị (gói plot), hình học giải tích (gói geometry), đại số tuyến tính (gói linalg),...

+ Thiết kế các đối tượng 3 chiều

+ v.v...

Tính toán các số lớn, các biểu thức cần độ chính xác cao

```
> 100!:
> 2^64:
> evalf(Pi,500):
```

Vẽ đồ thị các hàm số

```
> with(plots):
Warning, the name changecoords has been redefined
> with(plottools):
Warning, the assigned name arrow now has a global binding
> plot(x^3+4*x^2-1,x=-10..5,y=-10..15,thickness=2,numpoints=1000):
```

Tính đạo hàm, tích phân các hàm số

```
> diff(sin(2*x^2-1),x):
> int(sin(x)*cos(x),x):
```

Thiết kế các đối tượng 3 chiều

```
>tubeplot([10*cos(t),10*sin(t),0,t=0..2*Pi,radius=2*cos(7*t),numpoints=120,tubepoints=24],
scaling=CONSTRAINED):
>tubeplot({[10*cos(t),10*sin(t),0,t=0..2*Pi,radius=2*cos(7*t),numpoints=120,tubepoints=24]
,[0,10+5*cos(t),5*sin(t),t=0..2*Pi,radius=1.5,numpoints=50,
tubepoints=18]},scaling=CONSTRAINED):
```

BÀI 1. TÍNH TOÁN SỐ HỌC THÔNG DỤNG

1. Tính toán số học thông dụng

- Các phép toán số học: +, -, *, /
- Lũy thừa: ^, giai thừa: x!
- Logarit: ln(x), log[a](b), exp(x)
- Các hàm lượng giác: sin(x), cos(x), tan(x), cot(x),...
- Một số hàm khác: abs(x) - |x|, sqrt(x) - căn bậc 2 của x

```
> (-10+5^2)*(4-sqrt(36)):
> 99!:
> cot(Pi/4):
> 6!
```

2. Tính toán với độ chính xác theo yêu cầu

Lệnh evalf

- Cú pháp 1: evalf(bieu_thuc) - tính toán chính xác giá trị của biểu thức và biểu diễn kết quả với mặc định là 10 chữ số.
- Cú pháp 2: evalf(bieu_thuc, k) - tính toán chính xác giá trị của biểu thức và biểu diễn kết quả với k chữ số.

```
> 22/7:
> evalf(%):
> evalf(Pi,500):
```

3. Các thao tác với số nguyên tố

- Phân tích một số n thành thừa số nguyên tố: lệnh ifactor(n);
- Kiểm tra một số n có phải là số nguyên tố không?: lệnh isprime(n);
- Tìm số nguyên tố đứng sau một số n cho trước: lệnh nextprime(n);
- Tìm số nguyên tố đứng trước một số n cho trước: lệnh prevprime(n);
- Tìm ước số chung lớn nhất của 2 số nguyên dương a, b: lệnh gcd(a,b);
- Tìm bội số chung nhỏ nhất của 2 số nguyên dương a, b: lệnh lcm(a,b);
- Tìm số dư khi chia a cho b: lệnh irem(a,b);
- Tìm thương nguyên khi chia a cho b: lệnh iquo(a,b);

```
> ifactor(3000000000):
> ifactor(12233344445555566666677777777):
> gcd(157940,78864):
> lcm(12,15):
> prevprime(100):
> nextprime(100):
> nextprime(%):
> irem(145,7):
> iquo(145,7):
> y:=irem(145,7,'x'):
> x:
```

4. Giải phương trình nghiệm nguyên

Lệnh isolve:

- Cú pháp 1: isolve(phuong_trinh/he_phuong_trinh);
- Cú pháp 2: isolve(phuong_trinh/he_phuong_trinh, <danh_sach_tham_so>);

```
> isolve({x+y=36,2*x+4*y=100}):
> isolve(x+y=5,{a,b,c}):
```

5. Giải công thức truy hồi, giải dãy số

Lệnh rsolve:

- Cú pháp: rsolve(pt/he_pt_truy_hoi, ten_day_so);

```
> rsolve({f(n)=f(n-1)+f(n-2),f(0)=1,f(1)=1},f(n)):
> rsolve({f(n)=2*f(n-1)},f(n)):
> rsolve({g(n)=3*g(n/2)+5*n},g):
> rsolve(f(n)-f(n-1)=n^3,f):
> simplify(%):
> eqn:=f(n)=f(n-1)+4*n:
> rsolve(eqn,f):
> simplify(%):
```

6. Khái niệm biến số, hằng số

- Trong Maple, biến số được sử dụng thoải mái mà không cần khai báo, định nghĩa trước
- Biến số, hằng số được đặt tên thỏa mãn một số quy tắc sau:
  + Không bắt đầu bằng chữ số
  + Không chứa khoảng trắng và một số ký tự đặc biệt như: %,^,&,*,$,#,...
  + Không được trùng với tên một số hàm và lệnh của Maple: sin, cos, ln, min, max, ...
- Một biến số sẽ trở thành hằng số ngay khi nó được gán cho một giá trị nào đó.
- Nếu muốn biến một hằng số trở lại biến số, ta dùng phép gán: ten_bien:='ten_bien';

```
> isolve({x+y=36,2*x+4*y=100}):
> x:=2:
> isolve({x+y=36,2*x+4*y=100}):
> x:='x':
> isolve({x+y=36,2*x+4*y=100}):
```

7. Tính tổng và tích

Tính tổng: sử dụng lệnh sum (tính trực tiếp ra kết quả) hoặc Sum(biểu diễn dạng công thức)

Cú pháp: sum(bieu_thuc_trong_tong, bien :=gia_tri_dau .. gia_tri_cuoi);
Sum(bieu_thuc_trong_tong, bien :=gia_tri_dau .. gia_tri_cuoi);

Tính tích: sử dụng lệnh product (tính trực tiếp ra kết quả) hoặc Product (biểu diễn dạng công thức)

Cú pháp: product(bieu_thuc_trong_tong, bien :=gia_tri_dau .. gia_tri_cuoi);
Product(bieu_thuc_trong_tong, bien :=gia_tri_dau .. gia_tri_cuoi);

Lưu ý: giá trị vô cực được biểu diễn bằng từ khóa infinity

```
> Sum(x^2,x=1..5):
> value(%):
> sum(x^2,x=1..5):
> Sum(1/(x^2),x=1..infinity):
> value(%):
> Product((i^2+3*i-11)/(i+3),i=0..10):
> value(%):
> product((i^2+3*i-11)/(i+3),i=0..10):
```

BÀI 2. CÁC THAO CÁC ĐẠI SỐ CƠ BẢN

1. Khai triển, đơn giản và phân tích biểu thức đại số

Khai triển biểu thức đại số

- Cú pháp: **expand(bieu_thuc_dai_so);**

```
> expand(bt);
> bt:=(x+y)^15;
```

$$bt := (x + y)^{15}$$

```
> expand(bt);
```

$$\begin{aligned} & x^{15} + 15\,y\,x^{14} + 105\,y^2\,x^{13} \\ & + 455\,y^3\,x^{12} + 1365\,y^4\,x^{11} \\ & + 3003\,y^5\,x^{10} + 5005\,y^6\,x^9 \\ & + 6435\,y^7\,x^8 + 6435\,y^8\,x^7 \\ & + 5005\,y^9\,x^6 + 3003\,y^{10}\,x^5 \\ & + 1365\,y^{11}\,x^4 + 455\,y^{12}\,x^3 \\ & + 105\,y^{13}\,x^2 + 15\,y^{14}\,x \\ & + y^{15} \end{aligned}$$

Phân tích đa thức thành nhân tử

Cú pháp: **factor(bieu_thuc_dai_so);**

```
> factor(x^4-10*x^3+35*x^2-50*x+24):
```

Đơn giản biểu thức đại số

Cú pháp: **simplify(bieu_thuc_dai_so);**

```
> bt:=cos(x)^5+sin(x)^4+2*cos(x)^2-2*sin(x)^2-cos(2*x):
> simplify(bt):
```

Tối giản phân thức

Cú pháp: **normal(phan_thuc);**

```
> tu := x^3-y^3:
> mau := x^2+x-y-y^2:
> phanthuc := tu/mau:
> normal(phanthuc):
```

Thay giá trị cho biến trong biểu thức

Cú pháp: **subs(bien = gia_tri , bieu_thuc);**

```
> bt := x^2-1;
> subs(x=2,bt):
> bt := x^2-1;
```

$$bt := x^2 - 1$$

```
> subs(x=2,bt);
```

3

Chuyển đổi dạng biểu thức

Cú pháp: **convert(bieu_thuc, kieu_chuyen_doi);**

```
> bt:=(a*x^2+b)/(x*(-3*x^2-x+4)):
> convert(bt,parfrac,x):
> bt:=(x^2-1)/(x+2);
```

$$bt := \frac{x^2 - 1}{x + 2}$$

```
> convert(bt,parfrac);
```

$$x - 2 + \frac{3}{x + 2}$$

2. Định nghĩa hàm số

Cách 1: sử dụng toán tử ->

Cú pháp: **ten_ham := bien -> bieu_thuc_ham_so;**

```
> f := x->x^2+1/2:
> f(a+b):
```

Cách 2: sử dụng lệnh unapply

Cú pháp: **ten_ham := unapply(bieu_thuc, bien);**

```
> g:=unapply(x^3+2,x):
> g(4):
```

Định nghĩa hàm từng khúc

Cú pháp: **ten_ham := bien -> piecewise(đk_1, bt_1, đk_2, bt_2, ..., đk_n, bt_n);**

Ý nghĩa: nếu đk_i đúng thì hàm nhận giá trị là bt_i

```
> f:=x->piecewise(x<=-1,x^2-1,x<=1,-abs(x)+1,sin(x-1)/x):
> f(1):
```

3. Giải (bất) phương trình, hệ (bất) phương trình

Sử dụng một lệnh chung duy nhất: lệnh solve

- Cú pháp: **solve(phuong_trinh , {bien_1, bien_2, ...});**
  solve ({pt_1, pt_2, ...}, {bien_1, bien_2, ...});
  solve(bat_phuong_trinh , {bien_1, bien_2, ...});
  solve ({bpt_1, bpt_2, ...}, {bien_1, bien_2, ...});

```
> pt:=x^3-a*x^2/2+13*x^2/3=13*a*x/6+10*x/3-5*a/3:
> solve(pt,{x}):
> pt1:=abs((z+abs(z+2))^2-1)^2=9:
> solve(pt1,{z}):
> pt2:=(cos(x)-tan(x)=0):
> solve(pt2,{x}):
> pt3:=x^4-x^3+x^2-x+1:
> solve(pt3,{x}):
```

```
> hpt1:={x+y=36, x*4+y*2 = 100}:
> solve({x+y=36, x*4+y*2 = 100},{x,y}):
> solve((x-1)*(x-2)*(x-3) < 0, {x}):
> solve((x-1+a)*(x-2+a)*(x-3+a) < 0, {x}):
```

BÀI 3. VẼ ĐỒ THỊ VÀ CÁC VẤN ĐỀ LIÊN QUAN

1. Khởi tạo các hàm vẽ đồ thị

```
> with(plots):
```
Warning, the previous binding of the name arrow has been removed and it now has an assigned value
```
> with(plottools):
```

2. Vẽ đồ thị trong không gian 2 chiều Oxy

Vẽ đồ thị hàm thông thường:

Cú pháp: **plot(ham_can_ve, x=gt_dau..gt_cuoi, y=gt_dau..gt_cuoi, cac_tuy_chon);**
Một số tùy chọn thông dụng:
- Đặt màu cho đồ thị: **color = <màu>**
- Đặt độ dày k cho đồ thị: **thickness = k**
- Đặt số điểm vẽ cho đồ thị: **numpoints = k;**

```
> plot(x^3-3*x^2+1,x=-5..5,y=-5..5):
> f:=x->abs(x^3-x^2-2*x)/3-abs(x+1):
> plot(f(x),x=-5..5,y=-5..5):
```

Vẽ nhiều đồ thị trên cùng một hệ trục

Cú pháp: **plot([ham_1, ham_2,...], x=gt_dau..gt_cuoi, y=gt_dau..gt_cuoi, cac_tuy_chon);**
```
> plot([x^2,sin(x)],x=-2..2,color=[red,green]):
```

Vẽ đồ thị của hàm số không liên tục

Khi vẽ đồ thị của một hoặc nhiều hàm số có điểm gián đoạn, ta phải thêm tuy chọn **discont = true** để đồ thị được vẽ chính xác hơn
```
> g:=x->(x^2-1)/(x-2):
> plot(g(x),x=-10..10,y=-5..15,discont=true,color=blue):
```

Vẽ đồ thị hàm ẩn

Có những hàm số mà chúng ta không có được công thức tường minh y=f(x), khi đó để vẽ được đồ thị của chúng, ta sẽ dùng hàm implicitplot
Cú pháp: **implicitplot([bt_1, bt_2,...], x=gt_dau..gt_cuoi, y=gt_dau..gt_cuoi, cac_tuy_chon);**
```
> implicitplot(x^2/9+y^2/4=1,x=-4..4,y=-2..2):
> implicitplot(x^2-y^2-x^4=0,x=-1..1,y=-1..1):
```

Ứng dụng: vẽ đồ thị của hàm hữu tỷ

```
> f:=x->(x^2-1)/(x-2):
> bt:=convert(f(x),parfrac):
> tcx:=x->x+2:
> g1:=plot([f(x),tcx(x)],x=-10..10,y=-5..15,color=[blue,red],discont=true):
> g2:=implicitplot(x=2,x=-10..10,y=-5..15,color=green):
> display({g1,g2}):
```

3. Vẽ đồ thị trong không gian 3 chiều Oxyz

Vẽ đồ thị hàm thông thường

Cú pháp: **plot3d(ham_can_ve, x=gt_dau..gt_cuoi, y=gt_dau..gt_cuoi,z=gt_dau..gt_cuoi, cac_tuy_chon);**

```
> plot3d(x*exp(x^2),x=-2..2,y=-2..2,title="Do thi trong khong gian 3 chieu"):
> plot3d(-exp(-abs(x*y)/10)*sin(x+y)-cos(x*y),x=-Pi..Pi,y=-Pi..Pi,grid=[51,51]):
```

Vẽ đồ thị hàm ẩn

Cú pháp: **implicitplot3d(ham_can_ve, x=gt_dau..gt_cuoi, y=gt_dau..gt_cuoi,z=gt_dau..gt_cuoi, cac_tuy_chon);**

```
> implicitplot3d(x^2+y^2/4+z^2/9=1,x=-3..3,y=-3..3,z=-3..3):
```

4. Sự vận động của đồ thị

Cú pháp: **animate(ham_co_tham_so,x=gt_dau..gt_cuoi, tham_so = gt_dau..gt_cuoi);**
animate3d(ham_co_tham_so,x=gt_dau..gt_cuoi, y=gt_dau..gt_cuoi, tham_so = gt_dau..gt_cuoi);

Ý nghĩa: hiển thị sự biến đổi, vận động của đồ thị khi tham số thay đổi trong khoảng cho trước

```
> animate3d(cos(t*x)*sin(t*y),x=-Pi..Pi,y=-Pi..Pi,t=1..5):
> animate(t*x^2,x=-3..3,t=-5..5):
```

BÀI 4. GIỚI HẠN, ĐẠO HÀM, TÍCH PHÂN

1. Tính giới hạn

Cú pháp: **limit(ham_so,x=a);**
Limit(ham_so,x=a);
Ý nghĩa: tính giới hạn của ham_so khi x tiến đến a. Kết quả được thể hiện dưới dạng công thức (lệnh Limit) hoặc kết quả cụ thể (lệnh limit)

```
> f:=x->((sin(2*x))^2-sin(x)*sin(4*x))/x^4:
> Limit(f(x),x=0):
> value(%):
> limit(f(x),x=0):
```

Chú ý: muốn tính giới hạn của hàm số khi x tiến đến vô cực, ta chỉ việc thay a bằng từ khóa **infinity**.

```
> g := x->(2*x+3)/(7*x+5):
> Limit(g(x),x=infinity):
> value(%):
> limit(g(x),x=infinity):
```

Chú ý: muốn tính giới hạn của hàm số khi x tiến đến a từ bên trái hay bên phải, ta thêm vào một trong hai tùy chọn **left** hoặc **right**.

```
> h := x->tan(x+Pi/2):
> Limit(h(x),x=0,left):
> value(%):
> limit(h(x),x=0,right):
```

2. Tính đạo hàm

Tính đạo hàm cấp 1

Cú pháp: **diff(ham_so, bien);**
Diff(ham_so, bien);
Ý nghĩa: tính đạo hàm cấp 1 của ham_so theo bien. Kết quả được thể hiện dưới dạng công thức (lệnh Diff) hoặc kết quả cụ thể (lệnh diff)

```
> f := x->x^2*sqrt(x^2+1):
> Diff(f(x),x):
> value(%):
> diff(f(x),x):
> simplify(%):
```

Tính đạo hàm cấp cao

Cú pháp: **diff(ham_so, bien, bien, bien, ...);**
Diff(ham_so, bien, bien, bien, ...);
hoặc **diff(ham_so, bien$k);**
Diff(ham_so, bien$k);
Ý nghĩa: tính đạo hàm cấp k của ham_so theo bien. Kết quả được thể hiện dưới dạng công thức (lệnh Diff) hoặc kết quả cụ thể (lệnh diff)

```
> g := x->5*x^3-3*x^2-2*x^(-3):
> diff(g(x),x,x):
> h := x -> x^4 + x*sin(x):
> diff(h(x),x$2):
> simplify(%):
```

3. Tính tích phân

Tính tích phân xác định

Cú pháp: **int(ham_so, bien=a..b);**
Int(ham_so, bien=a..b);

Ý nghĩa: tính tích phân của ham_so với bien đi từ a đến b. Kết quả được thể hiện dưới dạng công thức (lệnh Diff) hoặc kết quả cụ thể (lệnh diff)

```
> f := x->1/(exp(x)+5):
> Int(f(x),x=0..ln(2)):
> value(%):
> evalf(%):
> g := x->cos(x)^2*cos(4*x):
> int(g(x),x=0..Pi/2):
```

Chú ý: ta có thể tính tích phân mở rộng khi a hay b có thể là vô cực (infinity)

```
> t := x->x/(x^4+1):
> int(t(x),x=0..infinity):
```

Tính tích phân bất định

Cú pháp: **int(ham_so, bien);**
Int(ham_so, bien);

Ý nghĩa: tính tích phân của ham_so theo bien. Kết quả được thể hiện dưới dạng công thức (lệnh Diff) hoặc kết quả cụ thể (lệnh diff)

```
> h := x->(3*x^2+3*x+3)/(x^3-3*x+2):
> t:=x->int(h(x),x):
> t(x):
```

4. Một số ứng dụng

Bài toán tính diện tích hình thang cong

*Bài 1. Tính diện tích hình thang cong giới hạn bởi hàm số f(x)=3*x-x^3 , trục Ox và hai đường thẳng x=0, x=1.*

```
> restart:
> with(plots):
Warning, the name changecoords has been redefined
> with(plottools):
Warning, the assigned name arrow now has a global binding
> f := x->3*x-x^2:
> g1:=plot(f(x),x=0..3,y=0..3,filled = true):
> g2:=implicitplot(x=3,x=0..4,y=0..3,color=blue):
> display({g1,g2}):
> int(f(x),x=0..3):
```

Bài 2. Tính diện tích hình thang cong giới hạn bởi hai hàm số f(x) = x^2 và g(x) = $x^{\frac{1}{2}}$

```
> f := x->x^2:
> g := x->sqrt(x):
> a:=solve(f(x)=g(x),x):
> plot([f(x),g(x)],x=a[1]..a[2]):
> abs(int(abs(f(x)-g(x)),x=a[1]..a[2])):
```

Bài toán khảo sát hàm số

Khảo sát và vẽ đồ thị hàm số $f(x) = \dfrac{\left(-x^2 + 3\cdot x - 3\right)}{2\cdot(x-1)}$

```
> f:=x->(-x^2+3*x-3)/(2*(x-1)):
> f1 := x->diff(f(x),x):
> f1(x):
> simplify(f1(x)):
> a:=solve(f1(x)=0,x):
> ct1:=a[1]:
> ct2:=a[2]:
> f(ct1):
> f(ct2):
> f2:=x->diff(f(x),x$2):
> f2(x):
> simplify(f2(x)):
> f(x):
> convert(f(x),parfrac,x):
> g1:=plot([-0.5*x+1,f(x)],x=-3..5,y=-2..4,color=[blue,red],discont=true):
> g2:=implicitplot(x=1,x=-3..5,y=-2..4,color=green):
> display({g1,g2}):
```

BÀI 5. HÌNH HỌC GIẢI TÍCH

1. Các tính toán trong hình học phẳng: gói geometry

Khởi tạo các hàm tính toán trong hình học phẳng

> with(geometry):

Các hàm trên đối tượng điểm

- Định nghĩa điểm: **point(ten_diem, hoanh_do, tung_do);**
- Hiển thị tọa độ của một điểm: **coordinates(ten_diem);**
- Xác định trung điểm đoạn thẳng tạo bởi hai điểm: **midpoint(ten_trung_diem, diem_1, diem_2);**

> point(A,2,3):
> point(B,-3,1):
> coordinates(A):
> coordinates(B):
> midpoint(M,A,B):
> coordinates(M):

Các hàm trên đối tượng đường thẳng

- Định nghĩa đường thẳng qua hai điểm:
  line(ten_dt, [diem_dau, diem_cuoi],[x,y]);
- Định nghĩa đường thẳng có phương trình cho trước:
  line(ten_dt,pt_duong_thang,[x,y]);
-Tìm giao điểm giữa hai đường thẳng:
  intersection(ten_giao_diem, dt_1, dt_2);
-Tìm góc giữa hai đường thẳng:
  FindAngle(dt_1, dt_2);
- Tính khoảng cách từ một điểm tới một đường thẳng:
  distance(diem, duong_thang);
- Xác định hình chiếu của một điểm lên trên một đường thẳng:
  projection(ten_hinh_chieu, diem, duong_thang);
- Xác định điểm đối xứng của một điểm qua một đường thẳng:
  reflection(ten_diem_dx, diem, duong_thang);

> line(d1,[A,B],[x,y]):
> line(d2,y=x+1,[x,y]):
> detail(d1):
> detail(d2):
> intersection(K,d1,d2):
> coordinates(K):
> FindAngle(d1,d2):
> distance(A,d1):
> distance(B,d2):
> projection(N,B,d2):
> coordinates(N):
> reflection(B1,B,d2):
> coordinates(B1):

Các hàm trên đối tượng đường tròn

- Định nghĩa đường tròn qua 3 điểm:

circle((ten_duong_tron,[diem1, diem2, diem3],[x,y]);

- Định nghĩa đường tròn có tâm và bán kính cho trước:

circle(ten_duong_tron,[tam, bk],[x,y]);

- Xác định bán kính đường tròn đã định nghĩa:

radius(tenduongtron);

- Xác định tọa độ tâm đường tròn đã định nghĩa:

coordinates(center(tenduongtron));

- Xác định diện tích đường tròn đã định nghĩa:

area(tenduongtron);

- Tìm tiếp tuyến với đường tròn tại một điểm:

tangentpc(tentieptuyen,diem,tenduongtron);

- Tìm tiếp tuyến với đường tròn qua một điểm:

tangentline(diem,tenduongtron,[tentieptuyen1, tentieptuyen2]);

```
> point(C,0,0):
> circle(c,[A,B,C],[x,y]):
> detail(c):
> radius(c):
> coordinates(center(c)):
> area(c):
> circle(c1,[C,5],[x,y]):
> detail(c1):
> tangentpc(t1,C,c):
> detail(t1):
> Equation(t1):
> TangentLine(t2,point(D,4,5),c,[l1,l2]):
```

Các hàm trên đối tượng tam giác

- Định nghĩa tam giác:

triangle(ten_tam_giac,[dinh1,dinh2,dinh3],[x,y]);

- Xác định diện tích tam giác:

area(ten_tam_giac)

- Xác định đường cao tam giác ứng với một đỉnh:

altitude(ten_duong_cao,dinh,ten_tam_giac);

- Xác định đường trung tuyến tam giác ứng với một đỉnh:

median(tenduongtrungtuyen,dinh,tentamgiac);

- Xác định đường phân giác tam giác ứng với một đỉnh:

bisector(ten_duong_phan_giac, dinh, ten_tam_giac);

- Xác định đường phân giác tam giác ứng với một đỉnh:

ExternalBisector(ten_duong_phan_giac,dinh,tentamgiac);

- Xác định trọng tâm tam giác:

centroid(ten_trong_tam,ten_tam_giac);

- Xác định trực tâm tam giác:

orthorcenter(ten_truc_tam, tentamgiac);

- Xác định đường tròn nội tiếp tam giác:

incircle(ten_duong_tron_noi_tiep,tentamgiac);

```
> triangle(ABC,[A,B,C],[x,y]):
> detail(ABC):
> area(ABC):
> altitude(ha,A,ABC):
> median(BM,B,ABC):
> detail(BM):
```

> bisector(Ct,C,ABC):
> detail(Ct):
> ExternalBisector(Cx,C,ABC):
> centroid(G,ABC):
> coordinates(G):
> orthocenter(H,ABC):
> coordinates(H):
> incircle(cc,ABC):
> detail(cc):

2. Các tính toán trong hình học không gian: gói geom3d

Khởi tạo

> with(geom3d):
Warning, these names have been rebound: AreCollinear, AreConcurrent, AreConjugate, AreParallel, ArePerpendicular, DefinedAs, Equation, FindAngle, GlideReflection, IsEquilateral, IsRightTriangle, OnSegment, RadicalCenter, altitude, area, center, centroid, coordinates, detail, distance, draw, dsegment, form, homology, homothety, intersection, inversion, line, midpoint, point, projection, radius, randpoint, reflection, rotation, segment, sides, translation, triangle, vertices
Warning, the assigned name polar now has a global binding

Các hàm trên đối tượng điểm

- Định nghĩa điểm: **point(ten_diem, hoanh_do, tung_do,cao_do);**
- Hiển thị tọa độ của một điểm: **coordinates(ten_diem);**
- Xác định trung điểm đoạn thẳng tạo bởi hai điểm: **midpoint(ten_trung_diem, diem_1, diem_2);**

> point(A,2,3,1):
> point(B,-3,1,3):
> coordinates(A):
> coordinates(B):
> midpoint(M,A,B):
> coordinates(M):

Các hàm trên đối tượng đường thẳng

- Định nghĩa đường thẳng qua hai điểm:
  line(ten_dt, [diem_dau, diem_cuoi]);
- Định nghĩa đường thẳng có phương trình tham so cho trước:
  line(ten_dt,pt_tham_so_duong_thang,ten_tham_so);
-Tìm giao điểm giữa hai đường thẳng:
  intersection(ten_giao_diem, dt_1, dt_2);
-Tìm góc giữa hai đường thẳng:
  FindAngle(dt_1, dt_2);
- Tính khoảng cách từ một điểm tới một đường thẳng:
  distance(diem, duong_thang);
- Xác định hình chiếu của một điểm lên trên một đường thẳng:
  projection(ten_hinh_chieu, diem, duong_thang);
- Xác định điểm đối xứng của một điểm qua một đường thẳng:
  reflection(ten_diem_dx, diem, duong_thang);

> line(d1,[A,B]):
> line(d2,[2+2*t,1-4*t,3*t],t):

```
> detail(d1):
Warning, assume that the parameter in the parametric equations is _t
Warning, assuming that the names of the axes are _x, _y, and _z
> detail(d2):
Warning, assuming that the names of the axes are _x, _y, and _z
> intersection(K,d1,d2):
intersection:   "the given objects do not intersect"
> FindAngle(d1,d2):
> distance(A,d1):
> distance(B,d2):
> projection(N,B,d2):
> coordinates(N):
> reflection(B1,B,d2):
> coordinates(B1):
```

Các hàm trên đối tượng mặt phẳng

- Định nghĩa mặt phẳng qua 3 điểm:
  plane(ten_mat_phang,[diem1, diem2, diem3],[x,y,z]);
- Định nghĩa mặt phẳng bằng phương trình tổng quát:
  plane(ten_mat_phang,pt_tongquat,[x,y,z]);
- Xác định giao tuyến của hai mặt phẳng:
  line(ten_giao_tuyen,[mp1,mp2]);
- Xác định khoảng cách giữa một điểm và một mặt phẳng:
  distance(ten_diem,ten_mat_phang);
- Xác định góc giữa hai mặt phẳng:
  FindAngle(ten_mp_1, ten_mp_2);

```
> point(C,0,0,0):
> plane(p,[A,B,C],[x,y,z]):
> detail(p):
> plane(p1,2*x-3*y+z=0, [x,y,z]):
> line(gt,[p,p1]):
> detail(gt):
Warning, assume that the parameter in the parametric equations is _t
> distance(A,p1):
> FindAngle(p,p1):
```