```
> with(plots):E:=implicitplot(y^2=x^3-5*x+3,x=-
> 5..5,y=-10..10,numpoints=1000):

> anim_elliptic:=proc()
> local P,a,b;
> P:=NULL;
> for a from -10 to 5 by 2  do
> for b from -10 to 5 by 2  do
> if (4*a^3+27*b^2)<>0 then
> P:=P,plots[implicitplot](y^2=x^3+a*x+b,x=-10..10,y=-10..10);
> fi;od;od;
> plots[display]([P],insequence=true,view=[-10..10,-10..10]);
> end:
> anim_elliptic();


> appart:=proc(a,b,P)
> local x,y;
> x:=P[1];y:=P[2];
> if y^2 <> x^3+a*x+b
> then RETURN(false);
> else RETURN(true);
> fi;
> end:
> appart(-36,0,[6,1]);
```

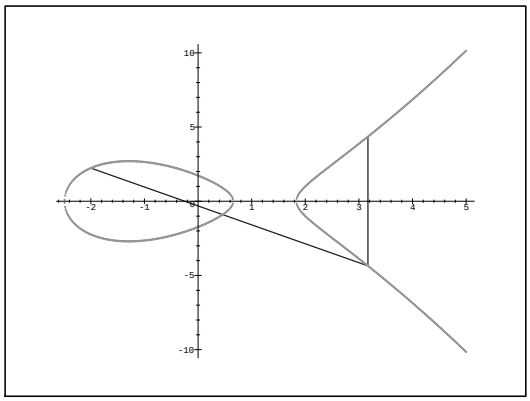
$$false$$

```
> somme := proc(P1,P2,a,b)
> local x1,x2,y1,y2,x3,y3,lambda,mu;
> x1:=P1[1];y1:=P1[2];x2:=P2[1];y2:=P2[2];
> if 4*a^3+27*b^2=0 then RETURN(`Delta nul`);fi;
> if  (P1=[0] or appart(a,b,P1)) and (P2=[0] or appart(a,b,P2))   then
> if P1 = [0] then RETURN(P2); fi;
> if P2 = [0] then RETURN(P1); fi;
> if x1 = x2 and y1 = -y2 then RETURN([0]);fi;
> if P1 = P2 then
> if y1 = 0 then RETURN([0]);fi;
> lambda := (3*x1^2+a)/(2*y1);
> else
> lambda := (y1-y2)/(x1-x2);
> fi;
> x3 := lambda^2-x1-x2;
> y3 := -y1 +lambda*(x1-x3);
> RETURN([x3,y3]):
> else RETURN(FAIL);
> fi;end:
> somme([-3,9],[-3,9],-36,0);
```

$$[\frac{25}{4}, \frac{-35}{8}]$$

```
> dessin_elliptic:=proc(P1,P2,a,b)
> local P3,P4,y2,E,x,L1;
> P3 := somme(P1,P2,a,b);
> if P1 = [0] or P2 = [0] or P3 = [0] then RETURN(FAIL);fi;
> P4 := [P3[1],-P3[2]];
> E := [];
> for x from -5 to 5 by 0.01 do
> y2 := evalf(x^3 + a*x + b);
> if y2 >= 0 then
> E := [op(E),[x,sqrt(y2)],[x,-sqrt(y2)]];
> fi;
> od;
> E := plot(E,style=point,color=green);
> L1 := plot([P1,P3,P4],color=blue,thickness=3);
> plots[display]([E,L1]);
> end:
> point_courbe:=proc(x,a,b)
> RETURN([(x),(sqrt(x^3+a*x+b))]);
> end:

> dessin_elliptic(point_courbe(-2,-5,3),point_c
> ourbe(-1,-5,3),-5,3);
```

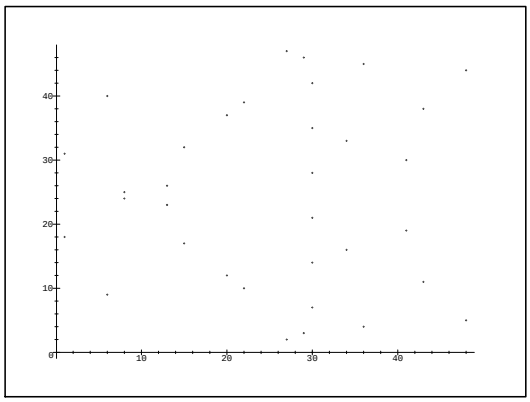


```
> sqrt_p:=proc(x,p)
> local i,racines;
> racines:=NULL;
> for i from 0 to p-1 do
> if i^2 mod p = x
> then racines := racines,i;
> fi;
> od;
> RETURN([racines]);
> end:

> sqrt_p(2,31);
```

$$[8, 23]$$

```
> dessin_elliptic_p:=proc(a,b,p)
> local y2,E,x,i;
> if (4*a^3+27*b^2) mod p = 0 then RETURN(‘Delta non nul‘);fi;
> E := [];
> for x from 0 to p-1 do
> y2 := (x^3 + a*x + b) mod p;
> for i in sqrt_p(y2,p) do
> E := [op(E),[x,i]];
> od;
> od;
> E := plot(E,style=point);
> RETURN(E);
> end:
> dessin_elliptic_p(26,3,49);
```

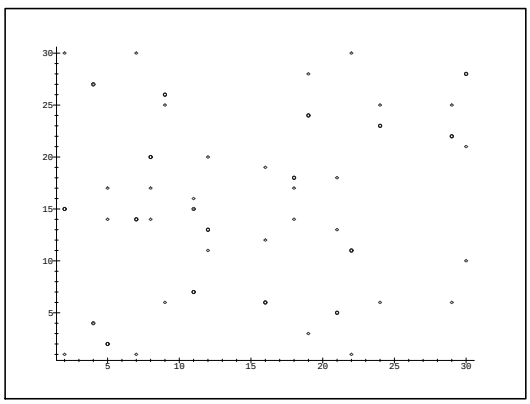


```
> appart_p:=proc(a,b,P,p)
> local x,y;
> x:=P[1];y:=P[2];
> if y^2 mod p <> (x^3+a*x+b) mod p
> then RETURN(false);
> else RETURN(true);
> fi;
> end:
```

```
> somme_p := proc(P1,P2,a,b,p)
> local x1,x2,y1,y2,x3,y3,lambda;
> if (4*a^3+27*b^2) mod p = 0  then RETURN(‘Delta nul‘);fi;
> if  (P1=[0] or appart_p(a,b,P1,p)) and (P2=[0] or appart_p(a,b,P2,p))
> then
> if P1 = [0] then RETURN(P2); fi;
> if P2 = [0] then RETURN(P1); fi;
> x1:=P1[1] mod p;y1:=P1[2] mod p;x2:=P2[1] mod p;y2:=P2[2] mod p;
> if x1 = x2 mod p and y1 = -y2 mod p then RETURN([0]);fi;
> if P1 = P2 then
> if y1 = 0 mod p then RETURN([0]);fi;
> if igcd(y1,p) <> 1 then RETURN([‘Diviseur‘,igcd(y1,p)]);fi;
> lambda := (3*x1^2+a)/(2*y1) mod p;
> else
> if igcd(x1-x2,p) <> 1 then
> RETURN([‘Diviseur‘,igcd(x1-x2,p)]);fi;
> lambda := (y1-y2)/(x1-x2) mod p;
> fi;
> x3 := (lambda^2-x1-x2) mod p;
> y3 := (-y1 +lambda*(x1-x3)) mod p;
> RETURN([x3,y3]):
> else RETURN(‘Mauvais points‘);
> fi;end:
> point_courbe_p:=proc(x,a,b,p)
> local X;
> X := x;
> while sqrt_p((X^3+a*X+b) mod p,p) = [] and X < p do
> X := X+1;
> od;
> RETURN([X,sqrt_p((X^3+a*X+b) mod p,p)[1]]);
> end:
> point_courbe_p(2,26,3,31);
```

$$[2,\ 1]$$

```
> dessin_elliptic_points_p:=proc(x1,x2,a,b,p)
> local y2,E,x,i,P1,P2,P,lambda;
> if (4*a^3+27*b^2) mod p = 0 then RETURN(‘Delta nul‘);fi;
> P1 := point_courbe_p(x1,a,b,p);
> P2 := point_courbe_p(x2,a,b,p);
> E := []; P:= [P1,P2];
> lambda := (P2[2]-P1[2])/(P2[1]-P1[1]) mod p;
> for x from 0 to p-1 do
> y2 := (x^3 + a*x + b) mod p;
> for i in sqrt_p(y2,p) do
> E := [op(E),[x,i]];
> P := [op(P),[x,(lambda*(x-P1[1])-P1[2]) mod p]];
> od;
> od;
> E := plot(E,style=point,symbol=diamond,color=red);
> P := plot(P,style=point,symbol=circle,color=blue);
> plots[display]([E,P]);
> end:
> dessin_elliptic_points_p(3,10,26,3,31);
```



```
> multiples_p:=proc(P,n,a,b,p)
> local i,Pm,Liste;
> Pm := [O];
> Liste := [O];
> for i from 0 to n-1 do
> Pm := somme_p(P,Pm,a,b,p);
> Liste := Liste,Pm;
> od;
> RETURN(Liste);
> end:

> multiples_p(point_courbe_p(1,26,3,31),33,26,3
> ,31);
```

[O], [2, 1], [16, 12], [22, 1], [7, 30], [11, 15], [12, 11], [18, 14], [5, 14], [29, 6], [19, 28], [24, 6], [21, 13], [9, 6], [30, 10], [8, 17], [4, 4], [4, 27], [8, 14], [30, 21], [9, 25], [21, 18], [24, 25], [19, 3], [29, 25], [5, 17], [18, 17], [12, 20], [11, 16], [7, 1], [22, 30], [16, 19], [2, 30], [O]

```
> multiples_p([15,10],8,1,-1,11*13);
```

[O], [15, 10], [100, 120], [94, 56], [57, 47], [31, 44], [68, 74], [61, 10], [67, 133]

```
> somme_p([-3,9],[-3,9],-36,0,201);
```

[*Diviseur*, 3]

```
> k_multiple_p:=proc(P,k,a,b,p)
> local K,M,Q,r;
> K:=k; M:=[O]; Q := P mod p;
> while K > 0 do
> r := K mod 2;
> if r = 1 then M := somme_p(M,Q,a,b,p); fi;
> if member(‘Diviseur‘,M) then RETURN(M);fi;
> Q := somme_p(Q,Q,a,b,p);
> if member(‘Diviseur‘,Q) then RETURN(Q);fi;
> K := iquo(K,2);
> od;
> RETURN(M);
> end:
> print(k_multiple_p([0,1],2,26,1,5411));
```

[*Diviseur*, 7]

```
> member(‘Diviseur‘,{‘Diviseur‘,5});
```

true

```
> ordre_point_p:=proc(P,a,b,p)
> local i;
> i:=1;
> while k_multiple_p(P,i,a,b,p) <> [O] do i:=i+1;od;
> RETURN(i)
> end:
> multiples_p(point_courbe_p(5,-37,0,11),5,-37,
> 0,11);
```

[O], [6, 4], [3, 2], [0, 0], [3, 9], [6, 7]

```
> ordre_point_p([0,0],26,0,31);
```

2

```
> multiples_p([0,0],6,-37,0,3);
```

[O], [0, 0], [O], [0, 0], [O], [0, 0], [O]

```
> friable := proc(n)
> local deux,trois,sept,onze,h;
> deux:=2;trois:=3;sept:=7;onze:=11;
> h:=rand(n);
> while deux <= h() do deux := deux*2;od;
> while trois <= h() do trois := trois*3;od;
> while sept <= h() do sept := sept*7;od;
> while onze <= h() do onze := onze*11;od;
> RETURN(deux*trois*sept*onze/462);
> end:
> h:=rand(6):h();
```

3

```
> friable(792);
```

645454656

```
> facto_p:=proc(n)
> local kP,a,b,P,B,s,k;
> if isprime(n) then RETURN(‘premier¡);fi;
> P:=[0,1];
> s := ceil(sqrt(n)):
> B:=s+1+2*ceil(sqrt(s));
> k:=friable(B);
> a:=1;b:=1;
> kP:=k_multiple_p(P,k,a,b,n);
> while not member(‘Diviseur‘,kP) do
> a := a+1;
> # k:=friable(B);
> kP:=k_multiple_p(P,k,a,b,n);
> od;
> printf(‘%a = %a * %a avec a = %a‘,n,kP[2],n/kP[2],a);
> end:

> facto_p(nextprime(123456789)*nextprime(987654
> 32));
12193264407559831 = 123456791 * 98765441 avec a = 49
```