

```

[ > #TP 3 (sem 2)
[ > #Exo 1 : Babyloniens
[ > baby := proc(a,n)
  local i,u;
  u:=1;
  if (n=0) then return u;
  else
  for i from 1 to n do
  u:=1/2*(u+a/u);
  end do;
  end if;
  return u;
  end proc;

baby := proc(a, n)
local i, u;
  u := 1;
  if n = 0 then return u else for i to n do u := 1 / 2*u + 1 / 2*a / u end do end if;
  return u
end proc
[ > evalf(baby(2,2),30);
                                     1.416666666666666666666666666667
[ > evalf(baby(2,3),30);
                                     1.41421568627450980392156862745
[ > evalf(baby(2,4),30);
                                     1.41421356237468991062629557889
[ > evalf(baby(2,5),30);
                                     1.41421356237309504880168962350
[ > evalf(baby(2,10),30);
                                     1.41421356237309504880168872421
[ > evalf(sqrt(2),30);
                                     1.41421356237309504880168872421
[ > #Exo 2 : Tchebychev
[ > tcheb := proc(n)
  local i,A,B,C;
  A:=1;
  B:=x;
  if n=0 then return A;
  else if n=1 then return B;
  else
  for i from 2 to n do
  C:=2*x*B - A;
  A:=B;
  B:=C;
  end do;

```

```

end if;
end if;
return C;
end proc;

```

```

>
>
>

```

```

tcheb := proc(n)

```

```

local i, A, B, C;

```

```

A := 1;

```

```

B := x;

```

```

if n = 0 then return A

```

```

else

```

```

    if n = 1 then return B

```

```

    else for i from 2 to n do C := 2*x*B - A; A := B; B := C end do

```

```

    end if

```

```

end if;

```

```

return C

```

```

end proc

```

```

> tcheb(0), tcheb(1), tcheb(2), tcheb(10);

```

```

1, x, 2 x2 - 1, 2 x (2 x (2 x (2 x (

```

```

2 x (2 x (2 x (2 x (2 x2 - 1) - x) - 2 x2 + 1) - 2 x (2 x2 - 1) + x) - 2 x (2 x (2 x2 - 1) - x)

```

```

+ 2 x2 - 1) - 2 x (2 x (2 x (2 x2 - 1) - x) - 2 x2 + 1) + 2 x (2 x2 - 1) - x)

```

```

- 2 x (2 x (2 x (2 x (2 x2 - 1) - x) - 2 x2 + 1) - 2 x (2 x2 - 1) + x) + 2 x (2 x (2 x2 - 1) - x)

```

```

- 2 x2 + 1) - 2 x (2 x (2 x (2 x (2 x (2 x2 - 1) - x) - 2 x2 + 1) - 2 x (2 x2 - 1) + x)

```

```

- 2 x (2 x (2 x2 - 1) - x) + 2 x2 - 1) + 2 x (2 x (2 x (2 x2 - 1) - x) - 2 x2 + 1)

```

```

- 2 x (2 x2 - 1) + x) - 2 x (2 x (

```

```

2 x (2 x (2 x (2 x (2 x2 - 1) - x) - 2 x2 + 1) - 2 x (2 x2 - 1) + x) - 2 x (2 x (2 x2 - 1) - x)

```

```

+ 2 x2 - 1) - 2 x (2 x (2 x (2 x2 - 1) - x) - 2 x2 + 1) + 2 x (2 x2 - 1) - x)

```

```

+ 2 x (2 x (2 x (2 x (2 x2 - 1) - x) - 2 x2 + 1) - 2 x (2 x2 - 1) + x) - 2 x (2 x (2 x2 - 1) - x)

```

```

+ 2 x2 - 1

```

```

> sort(expand (tcheb(10)));

```

```

512 x10 - 1280 x8 + 1120 x6 - 400 x4 + 50 x2 - 1

```

```

> eval(%,x=cos(a));

```

```

512 cos(a)10 - 1280 cos(a)8 + 1120 cos(a)6 - 400 cos(a)4 + 50 cos(a)2 - 1

```

```

> combine(%,trig);

```

```

cos(10 a)

```

```

> #Exo 3 :

```

```

> facto:=proc(n)

```

```

local i,s;

```

```
s:=1;
if n=0 then return s; else
for i from 1 to n do
s:=s*i;
end do;
end if;
return s;
end proc;
```

>

```
facto := proc(n)
```

```
local i, s;
```

```
    s := 1; if n = 0 then return s else for i to n do s := s*i end do end if; return s
```

```
end proc
```

```
> facto(0);
```

1

```
> facto(1);
```

1

```
> facto(2);
```

2

```
> facto(3);
```

6

```
> facto(4);
```

24

```
> facto(5), 5!;
```

120, 120

```
> factorec:=proc(n)
```

```
if (n=0) then return 1;
```

```
else return n*factorec(n-1);
```

```
end if;
```

```
end proc;
```

```
    factorec := proc(n) if n = 0 then return 1 else return n*factorec(n - 1) end if end proc
```

```
> factorec(0);
```

1

```
> factorec(1);
```

1

```
> factorec(2);
```

2

```
> factorec(3);
```

6

```
> factorec(4);
```

24

```
> factorec(5);
```

120

```

[ > # Exo 4 :
[ > fibo:=proc(n)
  local i,a,b,c;
  a:=0;
  b:=1;
  if n=0 then return a; else if n=1 then return b; else
  for i from 2 to n do
  s:=a+b;
  a:=b;
  b:=s;
  end do;
  end if;
  end if;
  return s;
  end proc;

```

Warning, `s` is implicitly declared local to procedure `fibo`

```

fibo := proc(n)
local i, a, b, c, s;
  a := 0;
  b := 1;
  if n = 0 then return a
  else
    if n = 1 then return b
    else for i from 2 to n do s := a + b; a := b; b := s end do
    end if
  end if;
  return s
end proc

```

```

[ > fibo(0);

```

0

```

[ > fibo(1);

```

1

```

[ > fibo(2);

```

1

```

[ > fibo(3);

```

2

```

[ > fibo(4);

```

3

```

[ > fibo(5);

```

5

```

[ > fibo(6);

```

8

```

[ > fibo(7);

```

```

[                                     13
[ > fibo(8);
[                                     21
[ > fibo(9);
[                                     34
[ > fibo(10);
[                                     55
[ > fiborec := proc(n)
[   if n=0 then return 0; else if n=1 then return 1; else
[   return fiborec(n-1)+fiborec(n-2);
[   end if; end if;
[   end proc;
[ fiborec := proc(n)
[   if n = 0 then return 0
[   else if n = 1 then return 1 else return fiborec(n - 1) + fiborec(n - 2) end if
[   end if
[ end proc
[ > fiborec(0);
[                                     0
[ > fiborec(1);
[                                     1
[ > fiborec(2);
[                                     1
[ > fiborec(3);
[                                     2
[ > fiborec(4);
[                                     3
[ > fiborec(5);
[                                     5
[ > fiborec(6);
[                                     8
[ > fiborec(7);
[                                     13
[ > fiborec(8);
[                                     21
[ > fiborec(9);
[                                     34
[ > fiborec(10);
[                                     55
[ >

```