Programmation et Structures de Données 1995-96
Corrigé du TD 1

Frédéric Vivien, bureau 306, e-mail: fvivien

22 septembre 1995

1 Premier contact

2 Les types de bases

```
#let c = 3;;
c : int = 3
#let d = 4;;
d : int = 4
#let e = c + d;;
e : int = 7
#let f = c * d;;
f : int = 12
```

3 Définitions

3.1 Définitions globales

3.2 Définitions locales

3.3 Liaisons statiques

```
#let x = 1 in
   let x = 2 in
      x ;;
- : int = 2
#let x = 1 in
   let x = x + 2 in
      x ;;
- : int = 3
#let x = 1 in
    let x = x + 2
    and y = x+ 1 in
       x + y;;
- : int = 5
```

4 Les fonctions

4.1 Fonctions d'une variable

4.2 Fonctions de plusieurs variables

4.3 Définitions locales de fonctions

4.4 Liaisons statiques

5 Vecteurs

6 Boucles For et conditionelles

7 Sortie de Caml

8 Exercices sur les polynômes

Un polynôme sera manipulé sous la forme du vecteur de ses coefficients.

```
#let v = [|4; 2; 0; 3|];;
v : int vect = [|4; 2; 0; 3|]

#let affiche_polynome v =
       for i = vect_length v - 1 downto 0 do
             print_int v.(i);
             print_char `x`;
             print_char `^`;
             print_int   i ;
             if i > 0 then
                begin
                print_char `+`
                end
       done;;
affiche_polynome : int vect -> unit = <fun>

#affiche_polynome v;;
3x^3+0x^2+2x^1+4x^0- : unit = ()
```

1.
```
#let affiche_polynome v =
       for i = vect_length v - 1 downto 0 do
           if v.(i) <> 0 then
               begin
               if v.(i) <> 1 then
                  begin
                  print_int v.(i)
                  end;
               if i <> 0 then
```

```
                  begin
                  print_char `x`;
                  if i <> 1 then
                     begin
                     print_char `^`;
                     print_int i
                     end ;
                  print_char `+`
                  end
               end
       done;;
affiche_polynome : int vect -> unit = <fun>
```

Ou mieux:

```
#let affiche_polynome v =
       for i = vect_length v - 1 downto 0 do
           if v.(i) <> 0 then
               begin
               if v.(i) <> 1 then
                  begin
                  print_int v.(i)
                  end;
               if i <> 0 then
                  begin
                  print_char `x`;
                  if i <> 1 then
                     begin
                     print_char `^`;
                     print_int i
                     end ;
                  if v.(i-1) <> 0 then
                     begin
                     print_string " + "
                     end
                  end
               end
            else
               if (i >= 1) & (v.(i-1) <> 0) then
                  begin
                  print_string " + "
                  end
       done;;
affiche_polynome : int vect -> unit = <fun>
```

2. Ré-écrire cette fonction en utilisant une fonction définie localement qui affiche un monôme.

```
#let affiche_polynome v =
        let affiche_monome coef degre =
            if coef <> 0 then
               begin
               if coef <> 1 then
                  begin
                  print_int coef
                  end;
               if degre <> 0 then
                  begin
                  print_char `x`;
                  if degre <> 1 then
                     begin
                     print_char `^`;
                     print_int degre
                     end ;
                  print_string " + "
                  end;
               end
        in
        for i = vect_length v - 1 downto 0 do
             affiche_monome v.(i) i;
        done;;
affiche_polynome : int vect -> unit = <fun>
```

3. Ecrire une fonction qui effectue la somme de deux polynômes.

```
#let somme u v =
    let s = make_vect (max (vect_length u) (vect_length v)) 0
    in
       for i = 0 to vect_length u - 1 do
           s.(i) <- u.(i)
       done;
       for i = 0 to vect_length v - 1 do
           s.(i) <- s.(i) + v.(i)
       done;
       s;;
somme : int vect -> int vect -> int vect = <fun>
```

4. Ecrire une fonction qui effectue le produit de deux polynômes.

```
#let produit u v =
    let p = make_vect (vect_length u + vect_length v -1 ) 0
    in
       for i = 0 to vect_length u - 1 do
          for j = 0 to vect_length v - 1 do
             p.(i+j) <- p.(i+j) + u.(i) * v.(j)
```

```
        done
     done;
     p;;

produit : int vect -> int vect -> int vect = <fun>
```