Module Lucas

1. Encore un Fibonacci vachement efficace qui utilise la suite de Lucas. Le principe est simple. La suite de Lucas est définie par $L_0 = 2$, $L_1 = 1$, $L_{n+1} = L_n + L_{n-1}$. Comme l'espace vectoriel des suites vérifiant $u_{n+1} = u_n + u_{n-1}$ est de dimension 2, il est facile de voir que F_n est combinaison linéaire de L_n et L_{n+1} ($F_n = (2L_n + L_{n+1})/5$).

Comme $L_n = \varphi^n + \overline{\varphi}^n$ où φ et $\overline{\varphi}$ sont le nombre d'or et son conjugué et puisque $\varphi\overline{\varphi} = -1$ on a $L_n^2 = L_{2n} + 2(-1)^n$ et $L_n L_{n+1} = L_{2n+1} + (-1)^n$.

2. On utilise la bibliothèque GNU MP.

```
open Gmp.Z
open Infixes
```

3. On introduit les deux constantes suivantes, pour améliorer la lisibilité.

```
let un = from_int 1
and deux = from_int 2
```

4. Pour éviter un enchevêtrement de if-then-else, on utilise un tableau de fonctions qui correspondent chacune à un cas de parité.

```
let cases =
  [|(function (ln, lnp1) ->
        ( sub_ui (ln *! ln) 2,
          sub_ui (ln *! lnp1) 1)) ;
    (function (ln, lnp1) ->
        ( sub_ui (ln *! lnp1) 1 ,
          add_ui (lnp1 *! lnp1) 2)) ;
    (function (ln, lnp1) ->
        ( add_ui (ln *! ln) 2,
          add_ui (ln *! lnp1) 1)) ;
    (function (ln, lnp1) ->
        ( add_ui (ln *! lnp1) 1,
          sub_ui (lnp1 *! lnp1) 2)) |]
```

5. L'application *lucas n* retourne la paire (L_n, L_{n+1}).

```
let rec lucas n =
  if n = 0
  then (deux, un)
  else
    let (ln, lnp1) = lucas (n lsr 1)
    in cases.(n land 3) (ln, lnp1)
```

6. Fonction principale qui prend n sur la ligne de commande et calcule F_n par la formule

$$F_n = \frac{2L_n + L_{n+1}}{5}.$$

```
let _ =
  let n = int_of_string (Sys.argv.(1)) in
  let (ln, lnp1) = lucas n in
    (tdiv_q_ui ((mul2exp ln 1) +! lnp1) 5)
```