```
(*************************************************)
(* Le module Frame définit les conventions d'appel *)
(*************************************************)

open Gen

type frame
(* Le type Frame.frame décrit les fonctions (sous-routines) du code intermédiair
e *)

val named_frame : string -> Pp.var_list -> Pp.type_expr option -> frame
   (* [named_frame f args r] crée un frame pour une fonction (ou procédure).

      [f] est le nom de la fonction.
      [args] est la liste de ses arguments.
      [r] est le type (optionnel) de son résultat.

      Deux labels sont créés pour représenter le point d'entrée et
      l'épilogue de la fonction.

      De nouveaux temporaires sont alloués pour stocker arguments et
      résultat de la fonction. *)
val frame_name : frame -> label
   (* renvoie le point d'entrée de la sous-routine *)
val frame_args : frame -> temp list
   (* renvoie la liste des temporaires choisis pour recevoir les arguments *)
val frame_result : frame -> temp option
   (* renvoie le temporaire choisi pour recevoir le résultat *)
val frame_return : frame -> label
   (* renvoie l'étiquette de l'épilogue, c'est-à-dire du code qui
      gère la fin de l'exécution de la fonction et le retour à
      l'appelant *)

val frame_size : frame -> int
    (* renvoie le nombre d'octets alloués à l'aide de [alloc_local] *)
val frame_size_label : frame -> string
    (* retourne le nom de la constante qui définit la taille du frame *)


(*********************************)
(* Allocation en pile (par mots)  *)
(*********************************)
val alloc_local : frame -> int
   (* alloue un mot au fond du frame, renvoie la position allouée sous
      forme d'offset (en octets) vis-à-vis de sp *)
val make_space_for_args : frame -> int -> unit
   (* au sommet du frame *)

(*********************************)
(* Allocation statique           *)
(*********************************)
val global_space : label
  (* étiquette de la zone (sert deux fois!) *)
val global_register : temp
  (* registre qui pointe sur la zone *)


(************************)
(* Frames des primitives *)
(************************)

val write_int : frame   (* Attend un entier *)
val writeln_int : frame (* Attend un entier *)
val read_int : frame    (* N'attend rien et renvoie un entier *)
val alloc  : frame      (* Attend un nombre de mots et renvoie une adresse *)
val bidon : frame       (* frame bidon, parfois utile *)

val wordsize : int
    (* taille d'un mot en octets *)
```