BEGGARI - BENGUEZZOU - CHATAIGNIER

PROJET
COMPILATION

TABLE DES MATIERES

COMPILATION

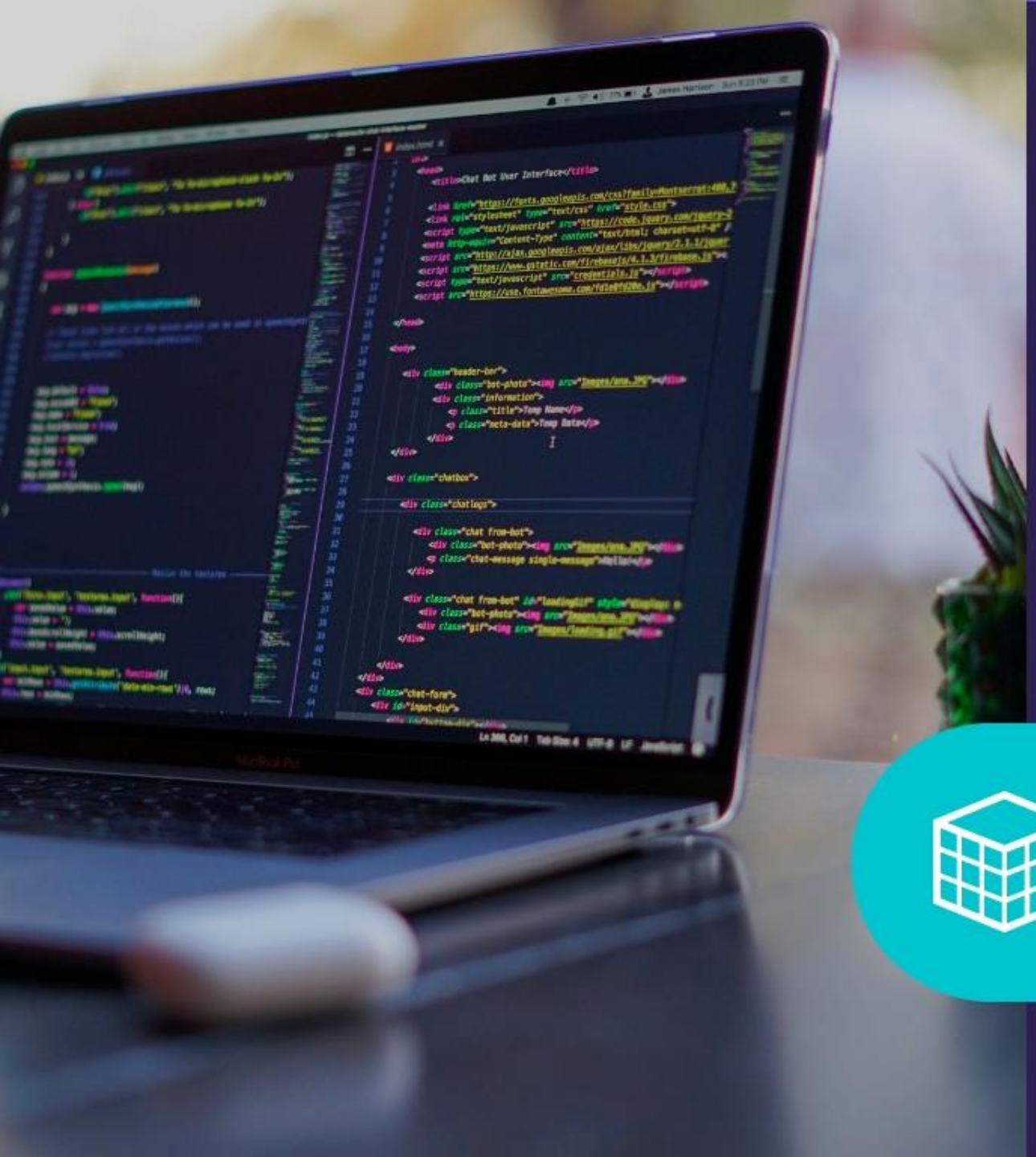
TABLE DES MATIERES

COMPILATION

TABLE DES MATIERES

EXECUTION

- Architecture
- TAD
  - pexec



COMPILATION

```
2     type date :
3         struct
4             jour : int;
5             mois : int;
6             annee : float
7         fstruct;
8     type personne :
9         struct
10            numero : int;
11            naissance : date;
12            age : int
13        fstruct;
14    type equipe : tableau[1..3, 1..2] de personne;
15    var p : personne;
16    var te : equipe;
17    var b : bool;
18    var i : float;
19    var j : int;
20    procedure a()
21        procedure b()
22        var T : int;
23
24        procedure c()
25        DEBUT
26        FIN;
27
28        procedure saisie()
```

INTRODUCTION

- CPYRR ?
- Comment lire un programme ?
- Analyses lexicale, synthaxique, sémantique
- Construction des tables
- Création du texte intermédiaire

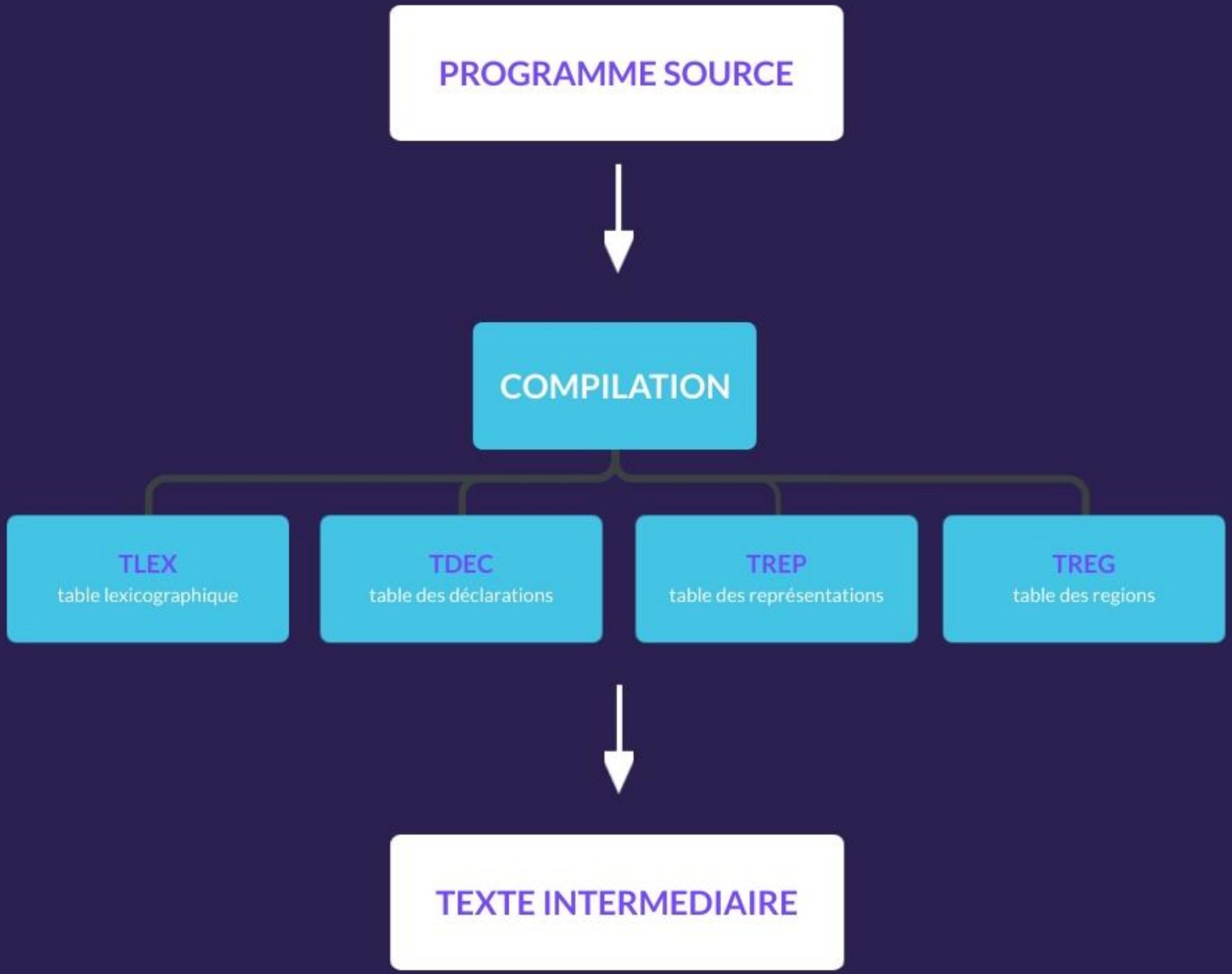
LEX et YACC

- Analyse du texte
- Envoie de TOKENS

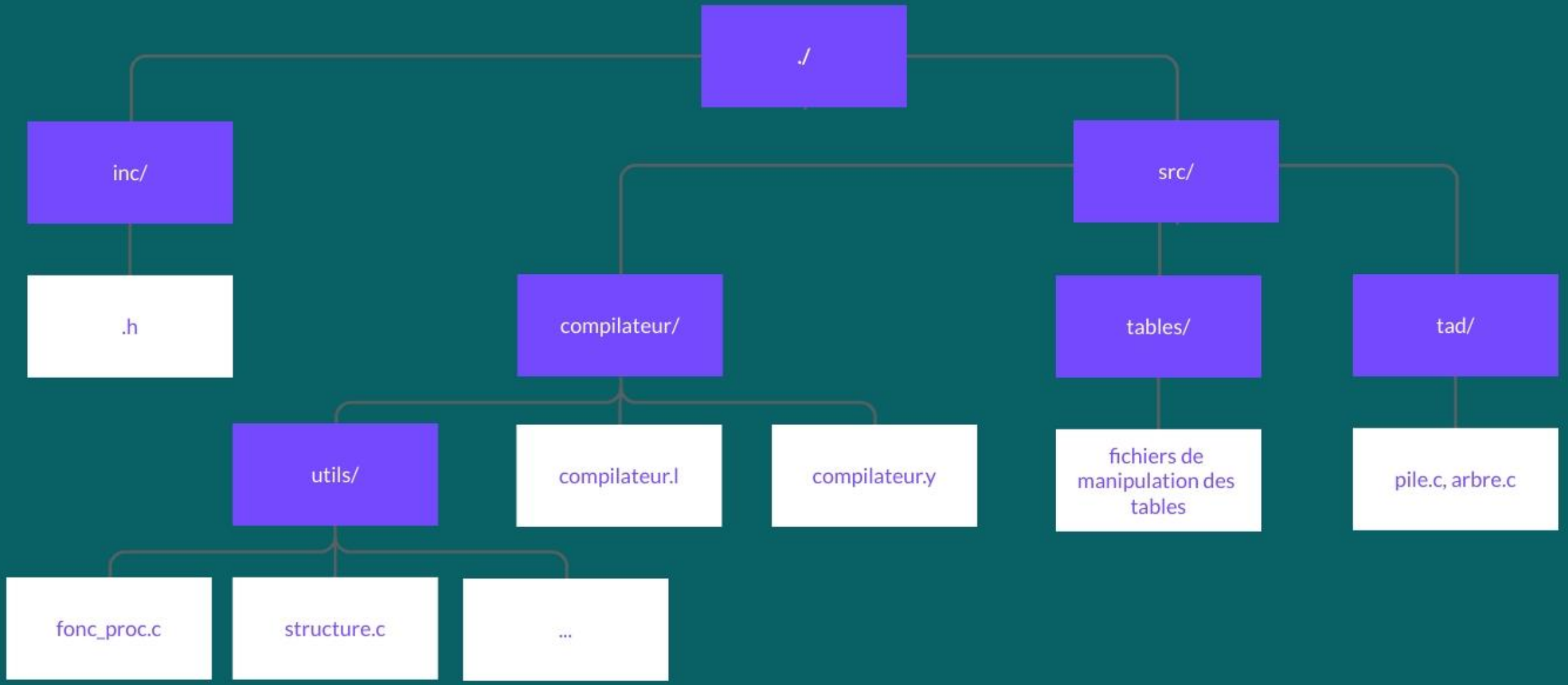
ANALYSE LEXICALE

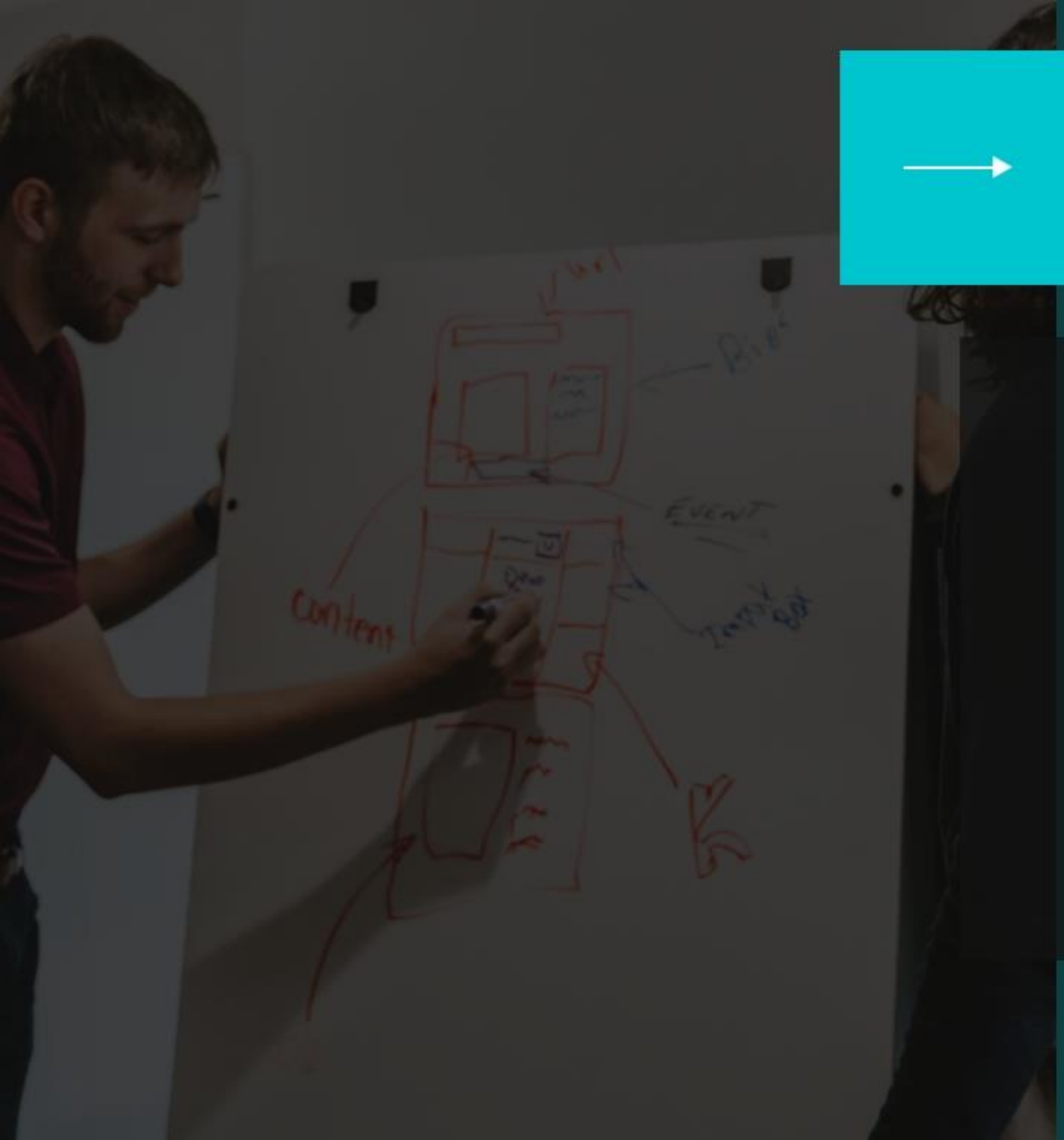
- Analyse de la cohérence
- Récupération de TOKENS

ANALYSE SYNTHAXIQUE


PROGRAMME SOURCE
COMPILATION
TLEX
table lexicographique
TDEC
table des déclarations
TREP
table des représentations
TREG
table des regions
TEXTE INTERMEDIAIRE

ARCHITECTURE





TAD

- arbre
- pile

pile

- p[0] : taille de la pile
- PILE_TMAX 50
- typedef int pile[PILE_TMAX]

- void pile_init(pile p);
- int pile_est_vide(pile p);
- void pile_empile(pile p, int num);
- int pile_depile(pile p);
- int pile_tete_de_pile(pile p);
- int pile_recupere_taille(pile p);
- int pile_est_dedans(pile p, int num);

arbre

- nature
- valeur_1
- valeur_2
- fils_gauche
- frere_droit

- `arbre arbre_creer_arbre_vide();`
- `int arbre_est_vide(arbre a);`
- `arbre arbre_creer_noeud(int nature, double valeur_1, double valeur_2);`
- `arbre arbre_creer_noeud_vide();`
- `arbre arbre_creer_arbre(int nature, double valeur_1, double valeur_2, arbre fils_gauche, arbre frere_droit);`
- `arbre arbre_concat_pere_fils(arbre pere, arbre fils);`
- `arbre arbre_concat_pere_frere(arbre pere, arbre fils);`
- `void arbre_affiche(arbre a);`

```
6|4|-1|mois
7|5|-1|annee
8|8|17|personne
9|6|-1|numero
10|9|-1|naissance
11|3|-1|age
12|6|16|equipe
13|1|-1|p
14|2|-1|te
15|1|-1|b
16|1|-1|i
17|1|-1|j
18|1|-1|a
19|1|-1|T
#FIN_TLEX
```

TABLES

- TLEX
- TDEC
- TREP
- TREG

TLEX

- taille du lexeme
- chainage vers le suivant
- lexeme

- `void tlex_init();`
- `void tlex_affiche();`
- `void tlex_ecrit(int index, int taille, int suivant, char *lexeme);`
- `int tlex_nouvelle_entree(char *lexeme);`

TDEC

- nature
- suivant
- region
- description
- execution

- `void tdec_init();`
- `void tdec_affiche();`
- `void tdec_ecrit(int index, int nature, int suivant, int region, int description, int execution);`
- `int tdec_nouvelle_entree(int tlex_index, int nature, int region, int description, int execution);`
- `void tdec_maj_taille_exec(int index);`
- `int tdec_recupere_taille_exec(int index);`
- `int tdec_recupere_region(int index);`
- `int tdec_trouve_index(int tlex_index, pile PREG);`
- `int tdec_trouve_index_fonction_procedure(int tlex_index, pile PREG);`

TREP

- valeur

- void trep_init();
- void trep_affiche();
- void trep_ecrit(int index, int valeur);
- int trep_nouvelle_entree(int valeur);
- void trep_maj_valeur(int index, int valeur);
- int trep_recupere_valeur(int index);

TREG

- taille
- nis
- arbre

- void treg_affiche();
- void treg_ecrit(int index, int taille, int nis);
- int treg_nouvelle_entree(int nis);
- int treg_recupere_taille(int index);
- void treg_maj_taille(int index, int taille);
- void treg_maj_arbre(int index, arbre a);

Association de nom

Analyse de fonction

- `static int tdec_trouve(int tlex_index, pile PREG);`

```
6|4|-1|mois
7|5|-1|annee
8|8|17|personne
9|6|-1|numero
10|9|-1|naissance
11|3|-1|age
12|6|16|equipe
13|1|-1|p
14|2|-1|te
15|1|-1|b
16|1|-1|i
17|1|-1|j
18|1|-1|a
19|1|-1|T
#FIN_TLEX
```

Insertion dans les tables

- variable
- tableau
- structure
- fonction / procedure

variable

- `void nouvelle_variable(int tlex_index, int tle_index_type);`

tableau

- taille
- trep_index_type
- trep_index_nb_dimensions
- tdec_index

- `void debut_nouveau_tableau();`
- `void nouvelle_dimension(int borne_min, int borne_max);`
- `void fin_nouveau_tableau(int tlex_index_type);`

structure

- trep_index_nb_champs
- tdec_index_type
- deplacement_exec

- `void debut_nouvelle_structure();`
- `void nouveau_champ(int tlex_index, int tlex_index_type);`
- `void fin_nouvelle_structure();`

fonction / procedure

- trep_index_type
- trep_index_nb_parametres
- tdec_index

- `void debut_nouvelle_fonction_ou_procedure(int type, int tlex_index);`
- `void nouveau_parametre(int tlex_index, int tlex_index_type);`
- `void fin_nouvelle_fonction_ou_procedure(int tlex_index, int tlex_index_type);`
- `void quitte_nouvelle_fonction_ou_procedure();`

Génération du texte intermediaire

Sauvegarde des tables et de l'arbre

- `void tlex_sauvegarde(FILE *f);`
- `void tdec_sauvegarde(FILE *f);`
- `void trep_sauvegarde(FILE *f);`
- `void treg_sauvegarde(FILE *f);`
- `void arbre_sauvegarde(FILE *f, arbre a, int niveau, int espace);`

Conclusion

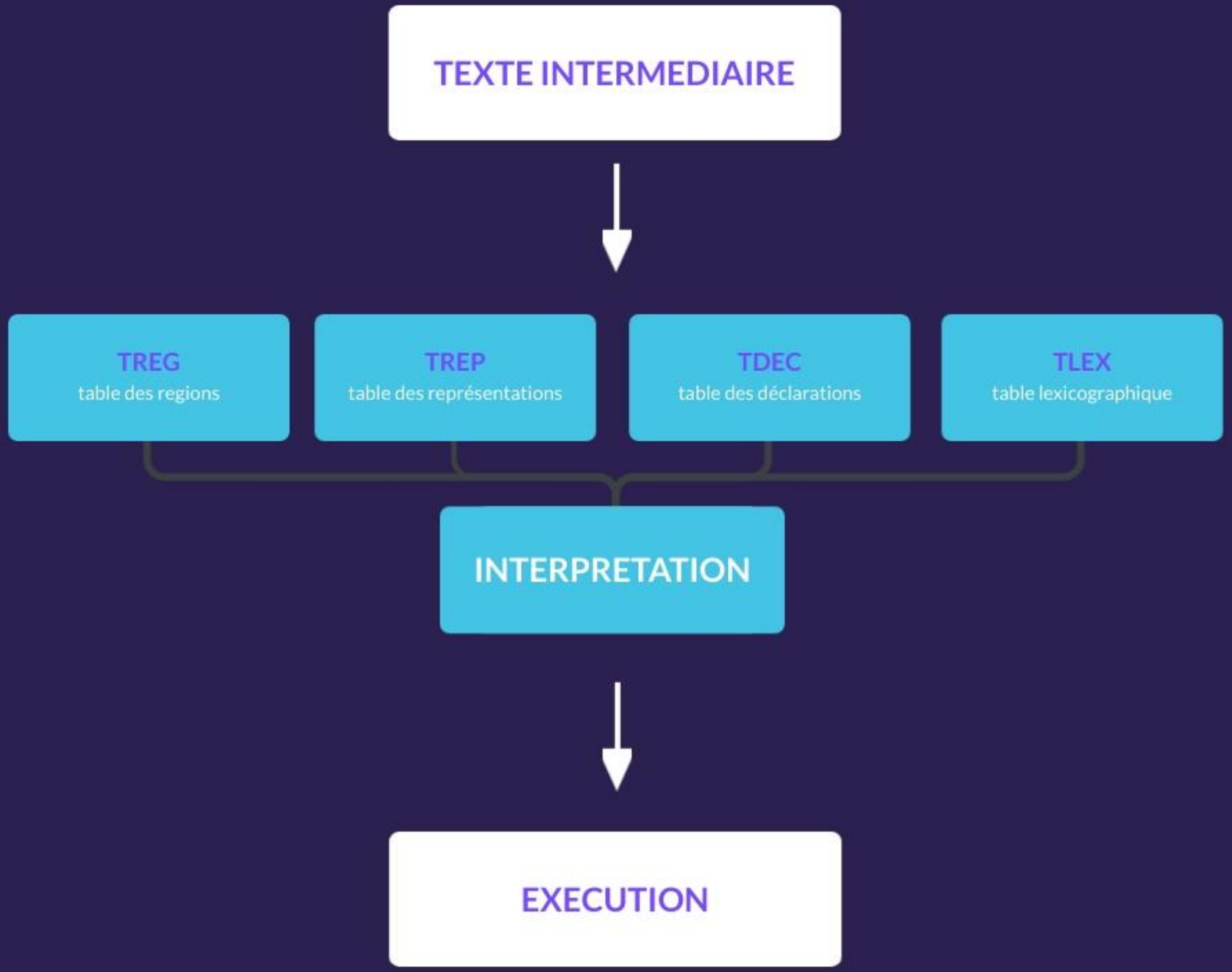
```
declaration_variable: VARIABLE idf DEUX_POINTS nom_type {
        int tlex_index_type = $4->valeur_1;
        nouvelle_variable($2, tlex_index_type);
        $$ = arbre_creer_noeud(A_DECL_VAR, $2, tlex_index_type);
    }
;
```

```
declaration_fonction: FONCTION idf { debut_nouvelle_fonction_ou_procedure(FONC, $2); } liste_parametres RETOURNE type_simple  {
        int tlex_index_type = $6->valeur_1;
        fin_nouvelle_fonction_ou_procedure(FONC, tlex_index_type);
    } corps {
        affecte_arbre_region_en_cours(arbre_concat_pere_fils(
            arbre_creer_noeud_vide(A_DECL_FONC),
            arbre_concat_pere_frere(
                $4,
                arbre_concat_pere_frere(
                    $6,
                    $8
                )
            )
        ));

        quitte_nouvelle_fonction_ou_procedure();

        $$ = arbre_creer_noeud(A_DECL_FONC, $2, VALEUR_NULL);
    }
;
```

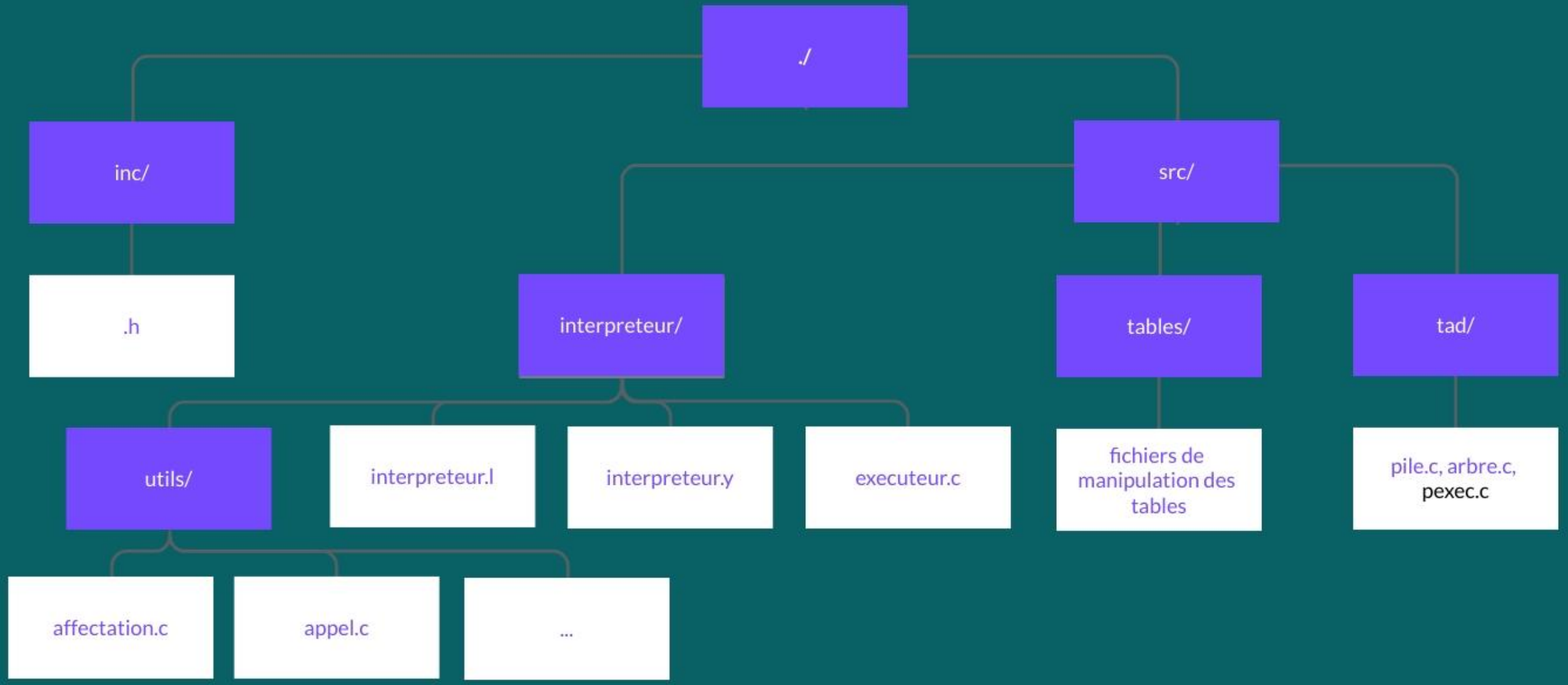
Conclusion

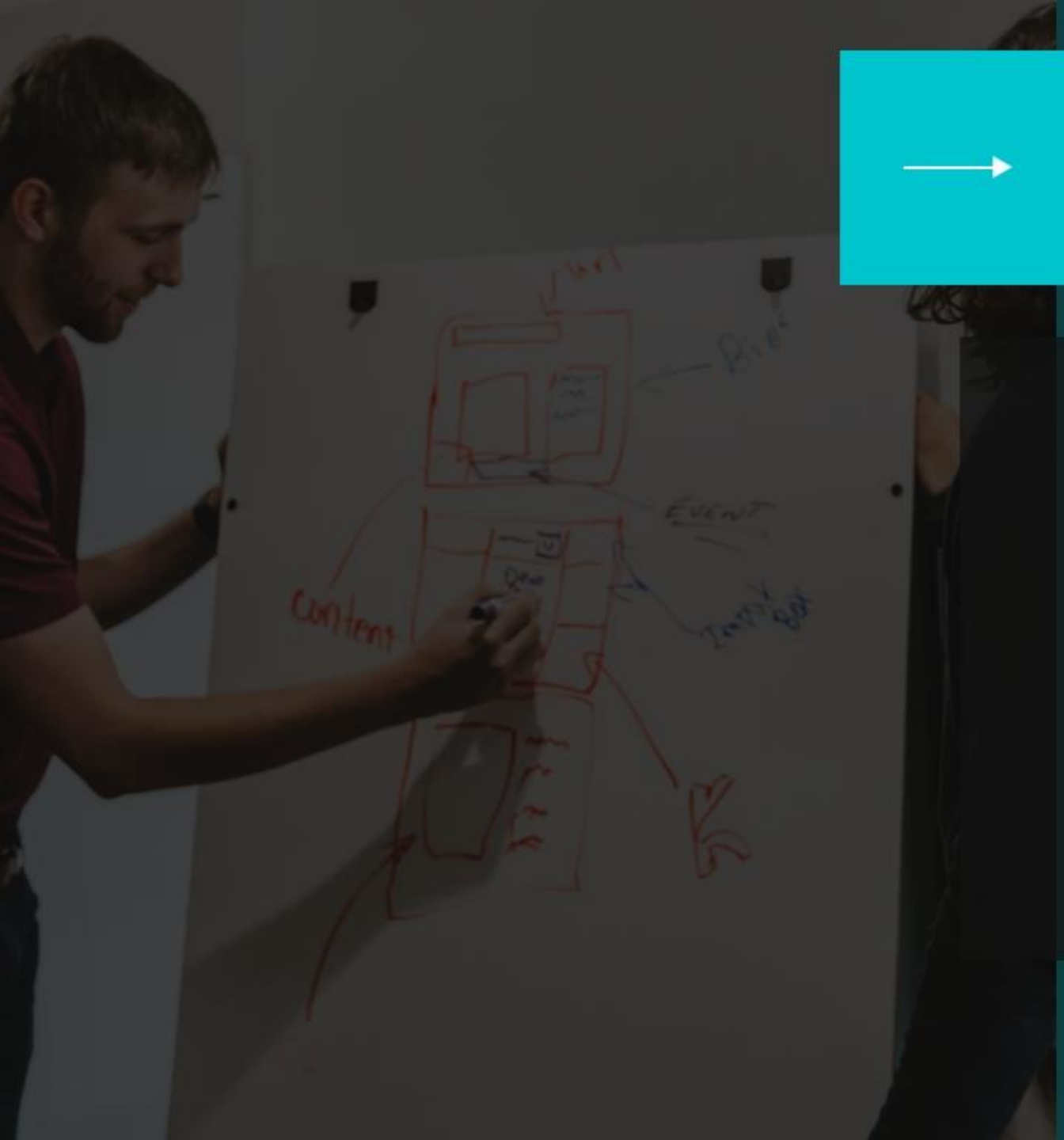
```
declaration_type: TYPE idf { maj_tlex_index($2); } DEUX_POINTS suite_declaration_type {
        $$ = arbre_concat_pere_fils(
            arbre_creer_noeud(A_DECL_TYPE, $2, VALEUR_NULL),
            $5
        );
    }
;

suite_declaration_type: STRUCT { debut_nouvelle_structure(); } liste_champs FSTRUCT {
        fin_nouvelle_structure();
        $$ = arbre_concat_pere_fils(
            arbre_creer_noeud_vide(A_STRUCT),
            $3
        );
    }
    | TABLEAU { debut_nouveau_tableau(); } dimension DE nom_type {
        fin_nouveau_tableau($5->valeur_1);
        $$ = arbre_concat_pere_fils(
            arbre_creer_noeud_vide(A_TAB),
            arbre_concat_pere_frere($3, $5)
        );
    }
;
```

EXECUTION


TEXTE INTERMEDIAIRE
TREG
table des regions
TREP
table des représentations
TDEC
table des déclarations
TLEX
table lexicographique
INTERPRETATION
EXECUTION

ARCHITECTURE





TAD

- pexec

pexec

- union cellule {
  int entier;
  float reel;
  int booleen;
  char caractere
  };
- PEXEC_TMAX 5000
- typedef cellule pexec[PEXEC_TMAX]

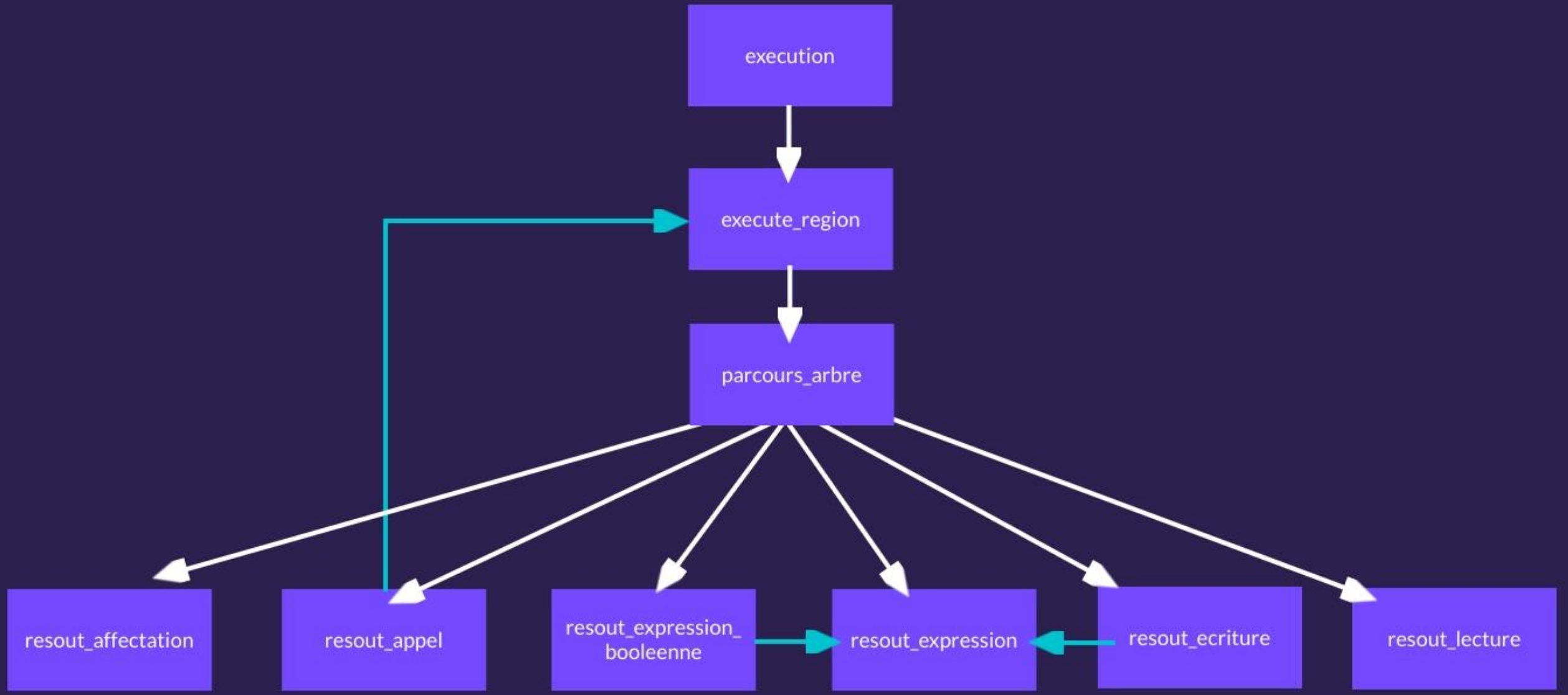
- cellule cellule_null();
- void pexec_empile(pexec p, cellule c, int *taille);
- void pexec_empile_entier(pexec p, int nb, int *taille);
- void pexec_empile_reel(pexec p, double nb, int *taille);
- void pexec_empile_booleen(pexec p, int bool, int *taille);
- void pexec_empile_caractere(pexec p, char carac, int *taille);
- cellule pexec_depile(pexec p, int *taille);
- void pexec_affiche(pexec p, int max);

```
6|4|-1|mois
7|5|-1|annee
8|8|17|personne
9|6|-1|numero
10|9|-1|naissance
11|3|-1|age
12|6|16|equipe
13|1|-1|p
14|2|-1|te
15|1|-1|b
16|1|-1|i
17|1|-1|j
18|1|-1|a
19|1|-1|T
#FIN_TLEX
```

Execution

- schéma simplifié de l'execution
- index dans PEXEC
- quelques exemples

Execution


execution
execute_region
parcours_arbre
resout_affectation
resout_appel
resout_expression_
booleenne
resout_expression
resout_ecriture
resout_lecture

Index dans PEXEC

- `int pexec_index_variable(int tlex_index, int deplacement_exec);`
- `int pexec_index_tableau(arbre a);`
- `int pexec_index_structure(arbre a);`

Quelques exemples

- ./exemples/representatifs/r-*.txt

BEGGARI - BENGUEZZOU - CHATAIGNIER

PROJET COMPILATION

CONCLUSION