République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

---

2ème année Cycle Supérieur (2CS)

Option : Systèmes Informatiques et Logiciels (**SIL1**)

Thème :

PROJET DE COMPILATION
Définition du langage

Réalisé par :

- **BOUNAB Chaima**
- **DJERFI Fatma**
- **MEKIRCHA Rafika Houda**
- **MELLITI Abdelmalek**

Encadré par : *ABDMEZIEM Riyadh*

Promotion 2025-2026

Table des matières

1 Définition générale du langage **1**

2 Structure générale du programme **1**

3 Commentaires **1**

4 Déclaration de variables simples et structurées **2**

4.1 Variables simples 2

4.2 Variables structurées : 3

1. Les enregistrements : 3
2. Les tableaux : 3
3. Les dictionnaires 4

5 Instructions de base **5**

5.1 Affectation 5

5.2 Condition 5

5.3 Boucles 6

5.4 Entrée / Sortie 6

6 Opérateurs logiques, de comparaison, et arithmétiques **7**

6.1 Opérateurs logiques 7

6.2 Opérateurs de comparaison 7

6.3 Opérateurs arithmétiques 7

7 Priorités et associativité **8**

7.1 Ordre de priorité 8

7.2 Associativité 8

1 Définition générale du langage

QueryLang est un langage impératif simple, inspiré de la syntaxe expressive de SQL et des langages procéduraux classiques, pour offrir une écriture claire, lisible et intuitive, en privilégiant une structure proche du langage humain.

2 Structure générale du programme

Un programme **QueryLang** commence par le mot-clé **BEGIN PROGRAM** suivi du nom du programme.

Le corps doit se terminer par l'instruction **END PROGRAM**.

```
BEGIN PROGRAM CalculNotes

   -- déclarations
   -- instructions

END PROGRAM;
```

Les instructions sont exécutées séquentiellement dans l'ordre d'écriture.

3 Commentaires

QueryLang supporte deux types de commentaires :

- **Commentaires sur une ligne:** Commençant par -- et se terminent en fin de ligne.

```
-- Ceci est un commentaire
```

- **Commentaires multi-lignes :** Délimités par /* */

```
/*
  Commentaire multi-ligne
 */
```

4 Déclaration de variables simples et structurées

4.1 Variables simples

➢ La syntaxe générale :

```
SET identifiant TYPE = valeur;
```

➢ Types supportés :

Type	Description
INTEGER	nombres entiers
STRING	chaînes de caractères
FLOAT	nombres réels
BOOLEAN	true / false

Exemple :

```
SET x INTEGER = 10;
SET message STRING = 'Bonjour';
SET prix FLOAT = 12.5;
SET actif BOOLEAN = true;
```

➢ Règles d'identifiants :

- Commencent par une lettre
- Contiennent lettres, chiffres, _
- Sensibles à la casse

4.2 Variables structurées :

1. Les enregistrements

➢ Syntaxe :

```
CREATE RECORD nom_enregistrement (
   champ1 type1,
   champ2 type2
);
```

➢ Initialisation :

```
CREATE RECORD Personne (
   nom STRING,
   age INTEGER,
   ville STRING
);
```

➢ Déclaration d'une instance :

```
SET p Personne;
p.nom = 'Ahmed';
p.age = 21;
```

2. Les tableaux

➢ Syntaxe :

```
SET nom ARRAY[TYPE, taille];
```

➢ Initialisation :

```
SET notes ARRAY[INTEGER, 5] = {12, 8, 15, 6, 18};
```

➢ Accès :

```
notes[0] = 20;
```

3. Les dictionnaires

➢ Syntaxe :

```
SET nom_dictionnaire DICTIONARY<type_clé, type_valeur>;
```

➢ Insertion :

```
contacts['Ali'] = 0658741234;
```

5 Instructions de base (affectation, conditions, boucles, entrée/-sortie,. . .etc.)

5.1 Affectation :

```
identifiant = expression;
```

Exemples :

```
x = x + 1;
message = 'OK';
```

5.2 Condition :

➢ Condition simple

```
WHEN x > 10
THEN
   PRINT 'Grand';
OTHERWISE
   PRINT 'Petit';
END WHEN;
```

La condition doit produire un booléen.

➢ Condition multiple (CASE)

```
CASE
   WHEN operation = '+' THEN
      resultat = a + b;
   WHEN operation = '-' THEN
      resultat = a - b;
   WHEN operation = '*' THEN
      resultat = a*b;
   WHEN operation = '/' AND b <> 0 THEN
      resultat = a / b;
   ELSE
      PRINT 'Opération invalide';
END CASE;
```

5.3 Boucles

➢ Boucle **WHILE**

```
LOOP WHEN x < 100
   x = x + 1;
END LOOP;
```

➢ Boucle **FOR**

```
LOOP ITERATE i FROM 0 TO 10
   PRINT i;
END LOOP;
```

5.4 Entrée / Sortie

➢ Affichage :

```
PRINT 'Premier nombre: ';
```

➢ Lecture :

```
INPUT a;
```

6 Opérateurs logiques, de comparaison, et arithmétiques

6.1 Opérateurs logiques

Opérateur	Fonction
AND	et logique
OR	ou logique
NOT	négation

6.2 Opérateurs de comparaison

Opérateur	Signification
=	égalité
<>	différent
>,<	comparaison
>=,<=	bornes

6.3 Opérateurs arithmétiques

Opérateur	Signification
+	addition
-	soustraction
*	multiplication
/	division
%	modulo

7 Priorites et Associativité

Ordre de priorité :

1. Parenthèses ()
2. NOT
3. Multiplication, division, modulo
4. Addition, soustraction
5. Comparaisons
6. Logiques AND, OR

Associativité :

1. Opérations arithmétiques : **gauche → droite**
2. Opérateurs logiques : **gauche → droite**
3. Affectation : **droite → gauche**