Introduction à la compilation :

Un compilateur est un programme qui traduit un autre programme

Traduction d’un programme source dans un langage cible :

* Analyser le programme conformément à la définition du langage source
* Produit d’éventuels messages d’erreurs
* Synthétise un code équivalent au programme source dans le langage cible

Interpréteurs :

Différent d’un compilateur par l’intégration de l’exécution et de la traduction.  
Utilisés pour les langages de commande

Les langages d’assemblage ou assembleurs sont des versions un peu plus lisibles du code machine avec des noms symboliques pour les opérations et pour les opérandes

⚠️ Chaque processeur à son langage d’assemblage

Partie frontale (front end)

Regroupe tout ce qui dépend du langage source plutôt que de la machine cible. On peut utiliser la même partie frontale sur une machine différente

Partie arrière (back end)

Regroupe le reste

Passes

Plusieurs phases peuvent être regroupées dans une même passe consistant à lire un fichier et écrire en un autre.

Analyse lexicale, syntaxique, sémantique et génération de code intermédiaire peuvent être regroupées en une seule passe.

Le cours concerne uniquement les langages impératifs (procéduraux)

LIPS et Prolog

La grammaire en compilation

Quatre composantes :

* Symboles terminaux : Sont atomiques (if, >=, ident, number)
* Symboles non terminaux : Sont dérivés en unité (Statement, Expr, Type)
* Productions : Règles donnant la décomposition des non terminaux
* Symbole de départ : Non terminal axiome (Depart)

Génération des instructions machine :

* Sélectionner les bonnes instructions
* Sélectionner les bons modes d’adressage

1957 - Fortran (Premiers compilateurs) (Expressions, instructions, procédures)

1960 – Algol (Première définition formelle d’un langage) (Grammaires sous forme de Backus-Naur, bloc, récursivité)

1970 – Pascal (Types, machines virtuelles (P-code)

1985 – C++ (Orientation objets, exceptions)

1995 – Java (Portabilité)