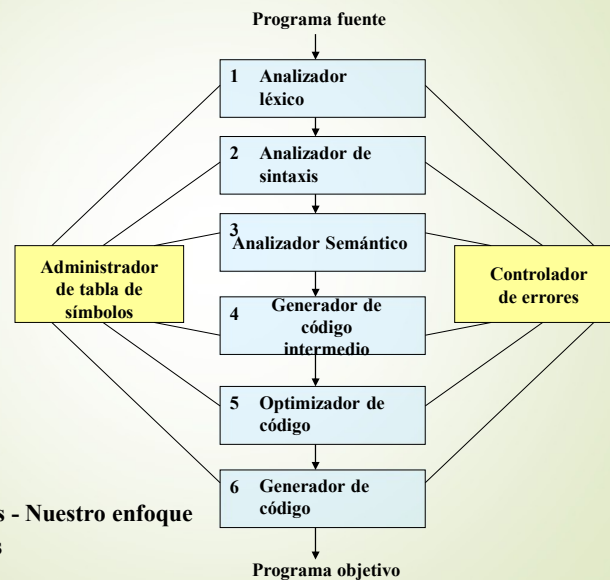


Las muchas Fases de un compilador



19

La tarea de síntesis para la compilación

- Generación de código intermedio
 - Versión del código independiente de la arquitectura para una máquina abstracta.
 - Fácil de producir y hacer la generación de código final dependiente de la máquina
- Optimización de código
 - Encontrar formas más eficientes de ejecutar código
 - Reemplazar código con instrucciones más óptimas
 - 2 enfoques: Lenguaje de alto nivel y optimización de "mirilla"
- Generación de código final
 - Generar código dependiente de máquina reubicable

20

Primos del compilador: Preprocesadores
 Proporcionan entrada a los compiladores

1. Procesamiento de macros

#define en C: Sustituye el texto antes de compilar

```
#define X 3
```

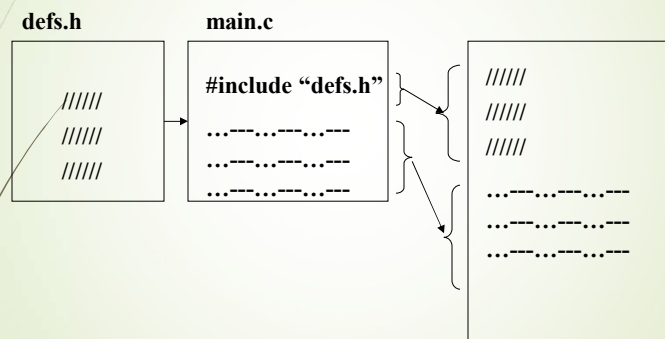
```
#define Y A*B+C
```

```
#define Z getchar()
```

21

2. Inclusión de archivos

#include en C - Trae otro archivo antes de compilar



22

3. Preprocesadores racionales

- Aumentan lenguajes "antiguos" con construcciones modernas
- Agregan macros para If - Then, While, Etc.
- #define Puede hacer que el código C sea más parecido a Pascal

```
#define begin {
#define end   }
#define then
```

23

4. Extensiones de idioma para un Sistema de base de datos

- QUEL - Lenguaje de consulta de base de datos incrustado en C.

```
## Retrieve (DN=Department.Dnum) where
##      Department.Dname = 'Research'
```

- se preprocesa en:

```
ingres_system("Retr...Research' ", ____, ____) ;
```

- Una llamada a procedimiento en un lenguaje de programación.

24