

Instituto Politécnico Nacional

Escuela Superior de Cómputo

Guía para el examen especial a título de suficiencia de Compiladores

CAPÍTULO 1 ARQUITECTURA DE LOS COMPILADORES E INTÉRPRETES

TEMAS EN FORMA DE LISTA

1. Definición de compiladores e intérpretes.
2. Arquitectura de los compiladores e intérpretes.
3. Máquinas virtuales.
4. Funcionalidad de las fases de un compilador.
 - a) Análisis léxico.
 - b) Análisis Sintáctico.
 - c) Análisis Semántico.
 - d) Generación de código intermedio.
 - e) Optimización de código.
 - f) Generación de código objeto.

TEMAS EN FORMA DE RENGLÓN

1.1 Definición de compiladores e intérpretes. 1.2 Arquitectura de los compiladores e intérpretes. 1.3 Máquinas virtuales. 1.4 Funcionalidad de las fases de un compilador. 1.4.1 Análisis léxico. 1.4.2 Análisis Sintáctico. 1.4.3 Análisis Semántico. 1.4.4 Generación de código intermedio. 1.4.5 Optimización de código. 1.4.6 Generación de código objeto.

LO QUE SE EVALUARÁ DEL CAPÍTULO 1
Los temas 1.1 y 1.2.

CAPÍTULO 2 CONSTRUCCIÓN DE ANALIZADORES LÉXICOS

TEMAS EN FORMA DE LISTA

1. Definición de clases léxicas por medio de expresiones regulares.
2. Creación de AFN (Autómatas finitos no deterministas) por medio de la construcción de Thompson.
3. Conversión de AFN a AFD (Algoritmo de subconjuntos).
4. Creación de un AFD a partir de una expresión regular.
5. Minimización de estados de un AFD.
6. Construcción de analizadores léxicos definidos por medio de expresiones regulares.
7. Generadores de analizadores léxicos (FLEX, JFLEX, JACCIE, GPLEX, etc.).

TEMAS EN FORMA DE RENGLÓN

2.1 Definición de clases léxicas por medio de expresiones regulares. 2.2 Creación de AFN (Autómatas finitos no deterministas) por medio de la construcción de Thompson. 2.3 Conversión de AFN a AFD (Algoritmo de subconjuntos). 2.4 Creación de un AFD a partir de una expresión regular. 2.5 Minimización de estados de un AFD. 2.6 Construcción de analizadores léxicos definidos por medio de expresiones regulares. 2.7 Generadores de analizadores léxicos (FLEX, JFLEX, JACCIE, GPLEX, etc.).

LO QUE SE EVALUARÁ DEL CAPÍTULO 2

Los temas 2.1, 2.2, 2.3, 2.4, 2.5, y 2.6.

CAPÍTULO 3 CONSTRUCCIÓN DE ANALIZADORES SINTÁCTICOS

TEMAS EN FORMA DE LISTA

1. Analizadores sintácticos.
 - a) Gramáticas ambiguas.
 - b) Eliminación de recursión izquierda en las gramáticas.
 - c) Árboles de sintaxis.
 - d) Definición de lenguajes libres del contexto.
2. Analizadores Sintácticos Descendentes.
 - a) Construcción de analizadores sintácticos por descenso recursivo.
 - b) Construcción de analizadores sintácticos LL(1).
3. Analizadores Sintácticos Ascendentes.
 - a) Analizadores LR(0).
 - b) Analizadores LR(1).
 - c) Analizadores LALR.
4. Generadores de analizadores sintácticos (YACC, JACC, JACCIE, GPPG).

TEMAS EN FORMA DE RENGLÓN

3.1 Analizadores sintácticos. 3.1.1 Gramáticas ambiguas. 3.1.2 Eliminación de recursión izquierda en las gramáticas. 3.1.3 Árboles de sintaxis. 3.1.4 Definición de lenguajes libres del contexto. 3.2 Analizadores Sintácticos Descendentes. 3.2.1 Construcción de analizadores sintácticos por descenso recursivo. 3.2.2 Construcción de analizadores sintácticos LL(1). 3.3 Analizadores Sintácticos Ascendentes. 3.3.1 Analizadores LR(0). 3.3.2 Analizadores LR(1). 3.3.3 Analizadores LALR. 3.4 Generadores de analizadores sintácticos (YACC, JACC, JACCIE, GPPG).

LO QUE SE EVALUARÁ DEL CAPÍTULO 3

Los temas 3.1.2, 3.1.4, 3.2.1, 3.2.2, y 3.3.1.

CAPÍTULO 4 GENERACIÓN DE CÓDIGO INTERMEDIO

TEMAS EN FORMA DE LISTA

1. Atributos.
 - a) Atributos heredados.
 - b) Atributos sintetizados.
2. Diseño de tabla de símbolos para soportar llamadas a funciones y/o procedimientos y ámbito de variables.
3. Generación de código de 3 y 4 direcciones.
4. Generación de código intermedio para estructuras iterativas y de control (for, while, if-else, switch).

5. Generación de código intermedio para el llamado a funciones y procedimientos.
6. Manejo del Polimorfismo en un compilador.

TEMAS EN FORMA DE RENGLÓN

4.1 Atributos. 4.1.1 Atributos heredados. 4.1.2 Atributos sintetizados. 4.2 Diseño de tabla de símbolos para soportar llamadas a funciones y/o procedimientos y ámbito de variables. 4.3 Generación de código de 3 y 4 direcciones. 4.4 Generación de código intermedio para estructuras iterativas y de control (for, while, if-else, switch). 4.5 Generación de código intermedio para el llamado a funciones y procedimientos. 4.6 Manejo del Polimorfismo en un compilador.

LO QUE SE EVALUARÁ DEL CAPÍTULO 4
El tema 4.3.

CAPÍTULO 5 OPTIMIZACIÓN DE CÓDIGO

TEMAS EN FORMA DE LISTA

1. Introducción a la optimización de código.
2. Análisis de flujo de datos.
3. Propagación de constantes.
4. Eliminación de redundancia parcial.
5. Ciclos en grafos de flujos.
6. Análisis basado en regiones.

TEMAS EN FORMA DE RENGLÓN

5.1 Introducción a la optimización de código. 5.2 Análisis de flujo de datos. 5.3 Propagación de constantes. 5.4 Eliminación de redundancia parcial. 5.5 Ciclos en grafos de flujos. 5.6 Análisis basado en regiones.

LO QUE SE EVALUARÁ DEL CAPÍTULO 5
No se evaluara nada.