컴파일러 설계 : **Overview**

2021, Spring

인천대학교 컴퓨터공학부

교수 홍 윤식

E-mail: yshong@inu.ac.kr

Text and Course Materials

Welcome to *컴파일러 설계*

- **The lecture opens for 30 years since 1991**
- **Text: Compiler Construction: Principles and Practice**
  - → **By Kenneth C. Louden, PWS Publishing Company, 1997**

- **Resources on the Web**
  - → **Lecture slides : PDF files**
  - → **Lab. handouts, homework**
  - → **previous exams**
  - → **You can get all of them from the web !**

  - → **LMS site: http://cyber.inu.ac.kr**

References – 컴파일러 이론 서적

- **Dragon book**
  - → **Compilers : Principles, Techniques, & Tools (2nd Ed.)**
    - **Alfred V. Aho,**
    - **Monica S. Lam,**
    - **Ravi Sethi,**
    - **Jeffrey D. Ullman, 2007 (1986년 초판의 개정판)**
  - → **Introduction to Automata Theory, Languages, and Computation**
    - **J. E. Hopcroft and J. D. Ullman, 1979**

References – 자연어 처리 서적

- 자연어 처리
  - → **nltk : https://www.nltk.org/**
  - → **konlpy : https://konlpy.org/en/latest/**

  - → 텐서 플로 **2**와 머신 러닝으로 시작하는 자연어 처리
    - 전 창욱, 최 태균, 조중현, 신성진, 위키북스, **2019.**
  - → 잡아라**!** 텍스트 마이닝 **with** 파이썬
    - 서대호, 비제이 퍼블릭, **2019.**
  - → **Natural Language Processing in Action**
    - **Hobson Lane, Hannes Max Hapke, Cole Howard, Manning, 2018**
  - → **Applied Text Analysis with Python**
    - **Benjamin Bengfort, Rebecca Bilbro, Tony Ojeda, O'Reilly, 2019.**
  - → **Natural Language Processing with Python Cookbook**
    - **Krishna Bhavsar, Naresh Kumar, Pratap Dangeti, Packt, 2017.**

A Compiler is a Translator

- A compiler *translates* a program you write
  - → … in a high-level language
    - C, C++, Java, etc.
  - → … into a low-level language
    - assembly language or machine language
  - → … that a computer *can understand* and eventually *execute*.

The process of language translation (1/4)

1! = 1, 2! = 2x1, 3!= 3x2x1, 4!= 4x3x2x1, …

1! = 1, 2! = 2x1!, 3!= 3x2!, 4!= 4x3!, … ➔ 순환 구조를 사용

수식으로 요약하면?

1. A person has an idea of how to compute something:

$$fact(n) = \begin{cases} 1 & \textbf{if } n \leq 0 \\ n \times fact(n-1) & \textbf{otherwise} \end{cases}$$

알고리즘으로 표현하면?

2. An algorithm captures the essence of the computation:

$$fact(n) = \textbf{if } n \leq 0 \textbf{ then } 1 \textbf{ else } n \times fact(n-1)$$

The process of language translation (2/4)

2. An algorithm captures the essence of the computation:

$$fact(n) = \textbf{if } n \leq 0 \textbf{ then } 1 \textbf{ else } n \times fact(n-1)$$

어떤 프로그래밍 언어를 사용할까?

3. The algorithm is expressed in some programming language:

```
int fact(int n) {
    if (n <= 0) return(1);
    else return(n*fact(n-1));
}
```

The process of language translation (3/4)

```
int fact(int n) {
    if (n <= 0) return(1);
    else return(n*fact(n-1));
}
```

W. Shakespeare 의 Sonnet(14줄의 시)

Shall I compare thee to a summer's day?
Thou art more lovely and more temperate:
Rough winds do shake the darling buds of May,
And summer's lease hath all too short a date:

당신을 한여름 어느 날과 비교해도 될까요?
물론 당신이 더 사랑스럽고 따스하겠지요.
뺨을 스치는 바람에 5월의 탐스런 꽃봉오리가 나부낍니다.
한여름 하루는 우리가 함께 보내기엔 너무 짧군요.

The process of language translation (4/4)

```
int fact(int n) {
   if (n <= 0) return(1);
   else return(n*fact(n-1));
}
```

Shall I compare thee to a summer's day?
Thou art more lovely and more temperate:
Rough winds do shake the darling buds of May,
And summer's lease hath all too short a date:

How does the machine knows it's seen a C program and not a Shakespeare sonnet?

How does the machine knows what is "meant" by the C program?

Finally …

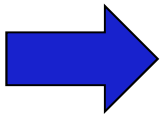
convenient for humans
to use and understand

Source Program

```
main() {
    int a;

    a += 5.0;
}
```

32 characters



low-level instruction set
of a computer

Target Program (Assembly)

```
_main:
    !#PROLOGUE# 0
    sethi   %hi(LF12),%g1
    add %g1,%lo(LF12),%g1
    save    %sp,%g1,%sp
    !#PROLOGUE# 1
    sethi   %hi(L2000000),%o0
    ldd [%o0+%lo(L2000000)],%f0
    ld  [%fp+-0x4],%f2
    fitod   %f2,%f4
    faddd   %f4,%f0,%f6
    fdtoi   %f6,%f7
    st  %f7,[%fp+-0x4]
```

24K bytes

Lecture Schedule

1주	컴파일러 입문
2주	어휘 분석 기초
3주	유한 오토마타
4주	어휘 분석 실습
5주	한글 처리 실습
6주	구문 분석 기초
7주	문맥 자유 문법
8주	중간 고사
9주	상향식 구문 분석
10주	하향식 구문 분석(1)
11주	하향식 구문 분석(2)
12주	자연어 처리 기초(1) – 정규 표현
13주	자연어 처리 기초(2) - 전처리
14주	자연어 처리 기초(3) – 문장 생성
15주	기말 고사

이 강의에서 무엇을 배울 수 있는가?

- 이론**(theory)**
  - → 정규 표현과 형식 언어**(CFG)**
    - 내가 언어를 정의할 수 있을까**?**
  - → 파싱 알고리즘 **: LL(k), LALR(k)**
  - → 한글 처리**(konlpy)**

- 응용**(application)**
  - → 자연어 처리 기초**(nltk** 기반**)**
    - **POS tagging,** 불용어 처리**,** 단어 원형 및 어근 찾기
    - **web scraping**
    - 유사도 분석**,** 문장 구분**,** 문장 생성

Basis for Grading

- **Exams 60%**
  - → **Midterm** **30%**
  - → **Final** **30%**

- **Presence** **20%**

- **Practices** **20%**
  - → 이론 수업 내용은 실습 수업에서 직접 실행시켜보며 확인
  - → 실습 예제는 **python**으로 **coding** 되어 있음