Introduction to Programming Language

Lecture 1

* Some materials adapted from ocw.mit.edu 6.0001

Today’s Agenda

- Course Layout
- What is computation
- Python

Course Layout

About this Course

- Instructor
  - Seongjin Lee
  - Email: insight@gnu.ac.kr
  - Office: 407-314
  - Office Hour: Every Thursday 11:00-12:00 or Make appointment
- Class
  - Time: Thursday 16:00-19:00
  - Place: 407-202
- Course webpage
  - http://open.gnu.ac.kr 컴퓨터프로그래밍기초

About this Course

- Python
  - 모두의 파이썬 20일 만에 배우는 프로그래밍 기초 | 이승찬 지음 | 길벗
  - Allen B. Downey, "Think Python: How to Think Like a Computer Scientist", 2015

- C
  - K.N. King, "C Programming: A Modern Approach," W. W. Norton & Company, 2nd Edition, 2008
  - Brian W. Kernighan, Dennis M. Ritchie, "C Programming Language" Prentice Hall, 1988 (2nd Edition) (Kernighan 의 C언어 프로그래밍)
  - Stephen Prata. "C Primer Plus". Sams, 2004 (C 기초 플러스)
  - 어서와 C언어는 처음이지! 초보자를 위한 C 프로그래밍 완벽가이드 | 그레그 페리 , 딘 밀러 지음 | 천인국 옮김 | 인피니티북스

About this Course

- Goal – Concepts, programming skills, problem solving
- Evaluation
  - Attendance – 10%
  - Assignments – 20%
  - Exam1 – 25%
  - Exam2 - 25%
  - Exam3 - 30%
  - Closed book and notes
  - Request for regrade within one week upon return; describe reasons in writing
    - what and why the score is incorrect or unfair
    - The written argument must be self-contained

About this Course

- Reading Assignments – Due before following class period

- Attendance
  - Participation is important part of this course
  - 3 absences without prior arrangement will lower your grade by one letter (each subsequent 1 absences will lower a grade by one letter)

- Fixed Sittings
  - If you have trouble hearing me or seeing the screen, now is the time to change your sits
  - Each students will be assigned a number, use that number on every piece of work you hand in

About this Course

- Academic Honesty
  - Assignments, quizzes, and exams done individually
  - No lying, cheating, copying
  - If found, no grade for that particular assessment
  - Suspicious work will be questioned thoroughly

About this Course

- No classes on 추석

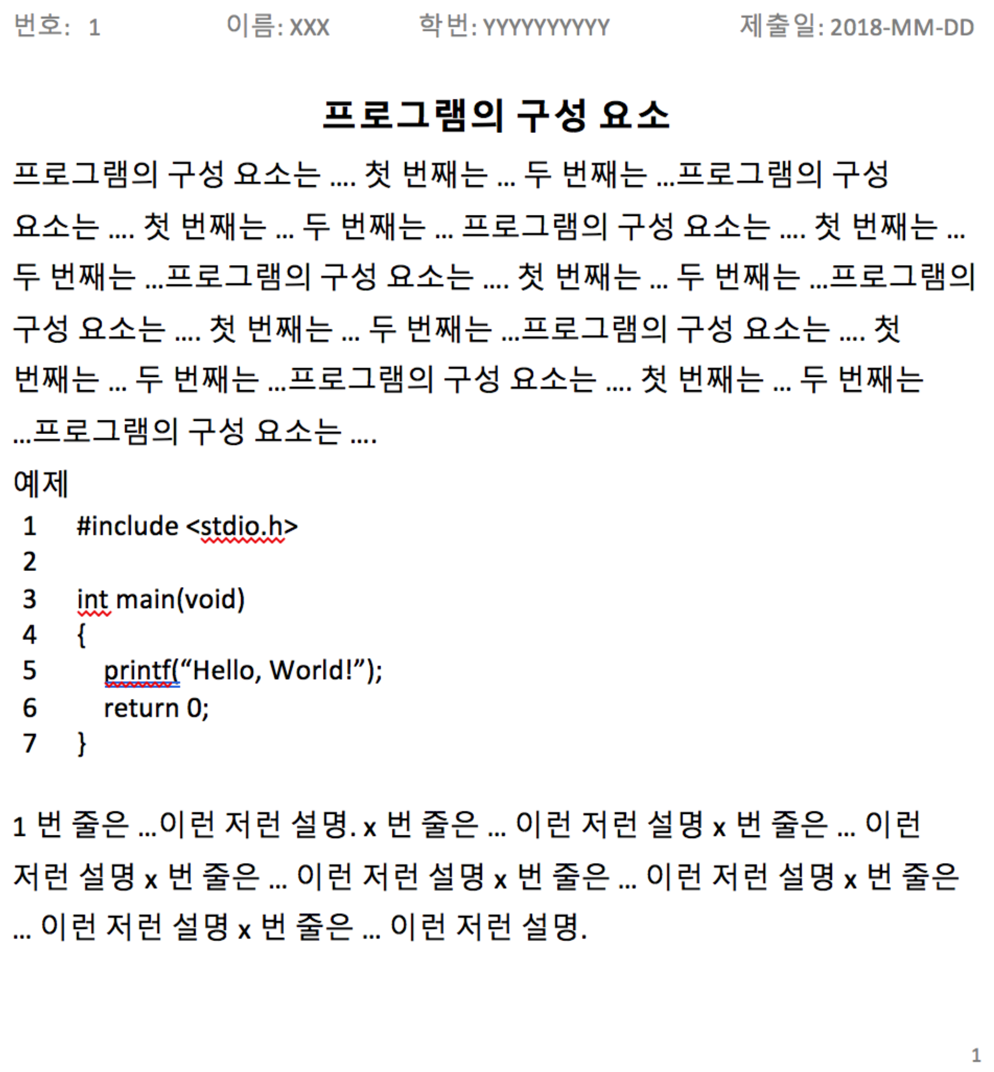
- Exams
  - Close book and notes
  - Exam1 on Oct. 10th (in class) – 407-101
  - Exam2 on Nov. 11th (in class) – 407-101
  - Exam3 on Dec. 12th (in class) – 407-101

- Survey
  - To give feedback on your understanding of material as well as help with material

Assignments for every week

- Choose any three concepts
  - Write the concept on the top of the page
  - Explain the concept with your words (make sure anybody can understand the concept)
  - Give an example of the concept
  - 1 page for each concept

- You are to hand it on every Tuesday morning
- No handwritten papers



번호: 1 이름: XXX 학번: YYYYYYYYYY 제출일: 2018-MM-DD

프로그램의 구성 요소

프로그램의 구성 요소는 .... 첫 번째는 ... 두 번째는 ...프로그램의 구성 요소는 .... 첫 번째는 ... 두 번째는 ... 프로그램의 구성 요소는 .... 첫 번째는 ... 두 번째는 ...프로그램의 구성 요소는 .... 첫 번째는 ... 두 번째는 ...프로그램의 구성 요소는 .... 첫 번째는 ... 두 번째는 ...프로그램의 구성 요소는 .... 첫 번째는 ... 두 번째는 ...프로그램의 구성 요소는 ....

예제

```
1   #include <stdio.h>
2
3   int main(void)
4   {
5     printf(“Hello, World!”);
6     return 0;
7   }
```

1 번 줄은 ...이런 저런 설명. x 번 줄은 ... 이런 저런 설명 x 번 줄은 ... 이런 저런 설명 x 번 줄은 ... 이런 저런 설명 x 번 줄은 ... 이런 저런 설명 x 번 줄은 ... 이런 저런 설명 x 번 줄은 ... 이런 저런 설명.

1

How to succeed in this course

- Read code and the manual
- Make mistakes and learn why
- Keep it simple

DON'T PRACTICE UNTIL YOU GET IT RIGHT

PRACTICE UNTIL YOU CAN'T GET IT WRONG

What is computation

What Does A Computer Do

- Fundamentally:
  - performs calculations
    - a billion calculations per second!
  - remembers results
    - 100s of gigabytes of storage!
- What kinds of calculations?
  - built-in to the language
  - ones that you define as the programmer
- Computers only know what you tell them

Types Of Knowledge

- declarative knowledge is statements of fact.

- imperative knowledge is a recipe or “how-to”.

A numerical Example

- square root of a number x is y such that y*y = x

- recipe for deducing square root of a number x (16)
  1. Start with a guess, g
  2. If g*g is close enough to x, stop and say g is the answer
  3. Otherwise make a new guess by averaging g and x/g
  4. Using the new guess, repeat process until close enough

g	g*g	x/g	(g+x/g)/2
3	9	16/3	4.17
4.17	17.36	3.837	4.0035
4.0035	16.0277	3.997	4.000002

What is a recipe

1. Sequence of simple steps
2. flow of control process that specifies when each step is executed
3. a means of determining when to stop

1 + 2 + 3 = an algorithm !

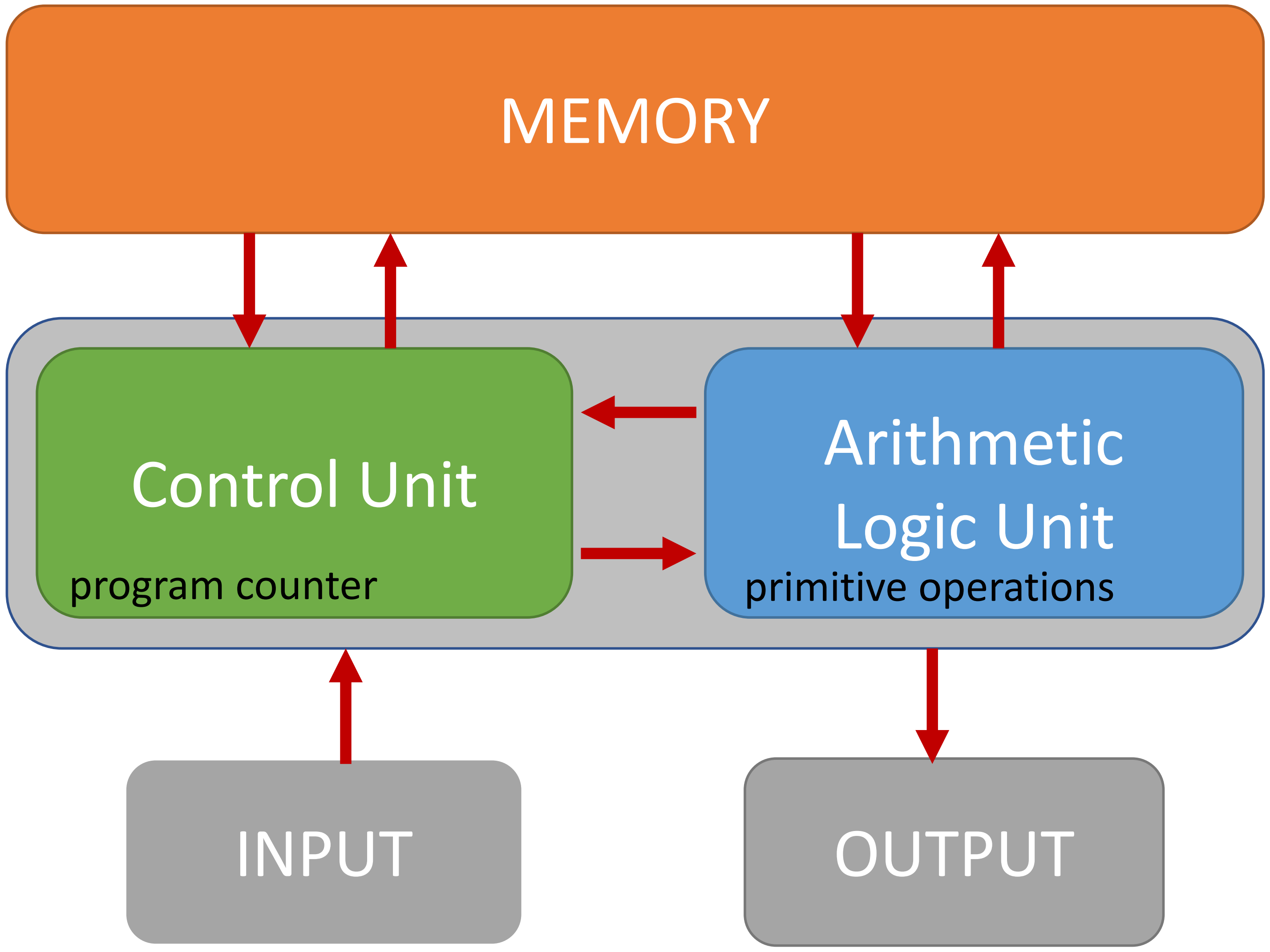
Computers are machines

- How to capture a recipe in a mechanical process

1. fixed program computer: Calculator

2. Stored Program computer: machine stores and executes instructions

Basic Machine Architecture



Stored Program Computer

- Sequence of instructions stored inside computer
  - built from predefined set of primitive instructions
    1. arithmetic and logic
    2. simple tests
    3. moving data
- special program (interpreter) executes each instruction in order
  - use tests to change flow of control through sequence
  - stop when done

Basic Primitives

- Turing showed that you can compute anything using 6 primitives
  - Move left, Move right, Print, Scan, Erase, Do nothing
  - if you are interested to learn more about it: reference or small video
- modern programming languages have more convenient set of primitives
- can abstract methods to create new primitives
- anything computable in one language is computable in any other programming language

Creating Recipes

- a programming language provides a set of primitive operations
- expressions are complex but legal combinations of primitives in a programming language
- expressions and computations have values and meanings in a programming language

Aspects of Languages

- Primitive constructs: Syntax and Semantic
  - Syntax: Defines the grammar
  - Semantic: is the meaning associated with syntactically correct symbols with no semantic errors

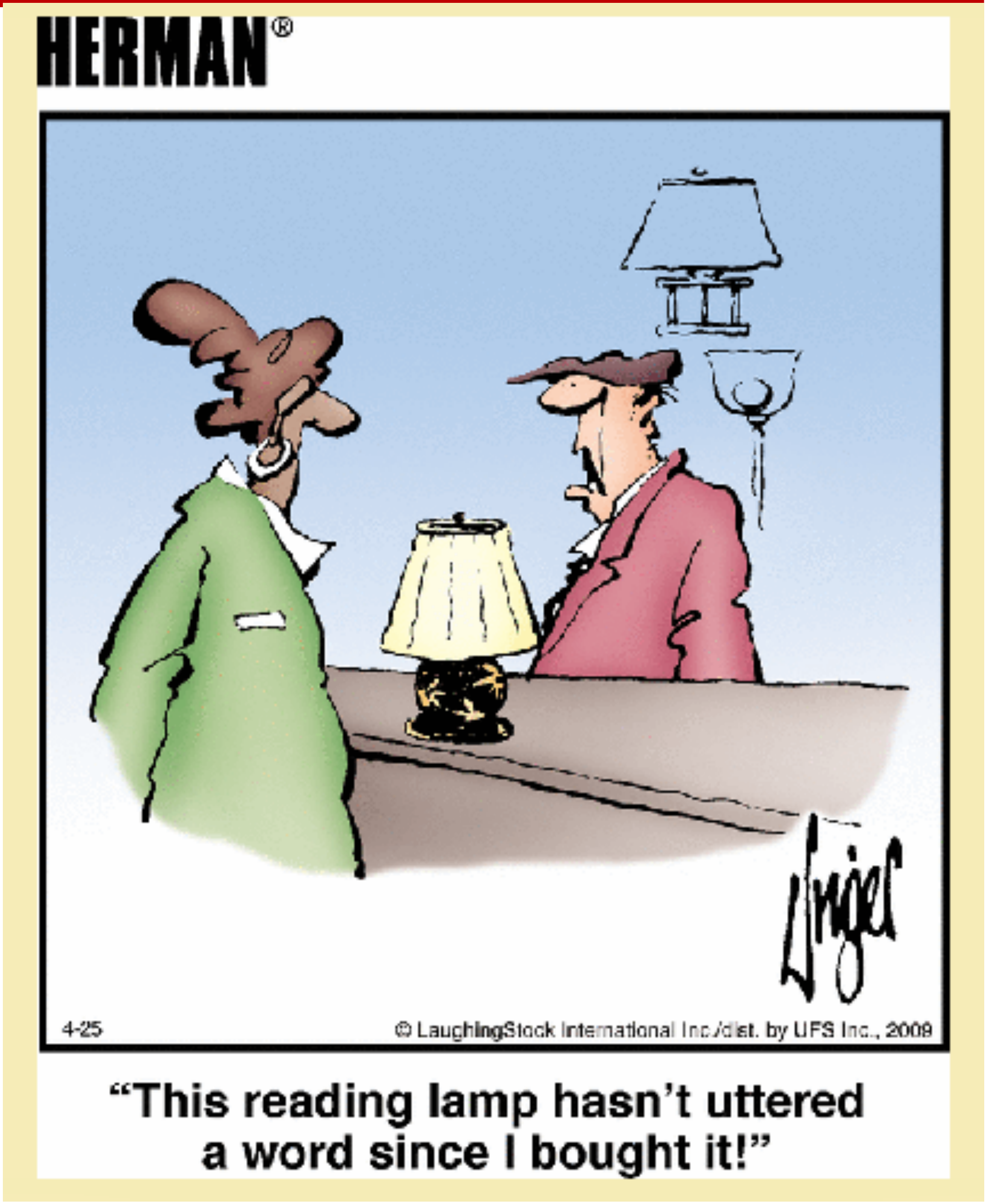
  - English/Korean: Words
    - 아버지 가방에 들어가신다 – syntactically valid but semantically not correct
    - 아기 고기 다리 – not syntactically valid

  - Programming Language: Numbers, Strings, Simple operators
    - 3.14*8 – syntactically valid
    - “hi”5 – not syntactically valid

Aspects of Languages

- Natural languages have many meanings

- Programming Languages have only one meaning but may not be what programmer intended


HERMAN®
4-25
© LaughingStock International Inc./dist. by UFS Inc., 2009
"This reading lamp hasn't uttered a word since I bought it!"

Where things go wrong

- syntactic errors
  - common and easily caught
- static semantic errors
  - some languages check for these before running program
  - can cause unpredictable behavior
- no semantic errors but different meaning than what programmer intended
  - program crashes, stops running
  - program runs forever
  - program gives an answer but different than expected

Python

Python Programs

- a program is a sequence of definitions and commands
  - definitions evaluated
  - commands executed by Python interpreter in a shell
- commands (statements) instruct interpreter to do something
- can be typed directly in a shell or stored in a file that is read into the shell and evaluated

Objects

- Python program manipulates data objects
- Objects have a type that defines the kinds of things program can do to them
- objects are
  - scalar (cannot be subdivided)
  - non-scalar

Scalar objects

- int – represent integers, ex. 1, 2, 3, 4, etc.
- float – represent real numbers, ex. 3.14, 48.12
- bool – represent Boolean values True and False

- can use type() to see the type of an object

```
>>> type(5)
int
>>> type(3.14)
float
```

Type conversion (cast)

- can convert object of one type to another

- example:
  - float(3) converts integer 3 to float(3.0)
  - int(3.9) truncates float 3.9 to integer 3

Printing to console

- to show output from code to a user, use print command

```
In    [30]: 3+8
Out  [30]: 11
```

“out” tells you it’s an interaction within the shell only

```
In     [31]: print(3+8)
11
```

No “out” means it is actually shown to a user when you run a file

Expressions

- combines objects and operators to form expressions
- an expression has a value, which has a type
- syntax for a simple expression
  - <object> <operator> <object>

Operators for int and float types

- i+j → sum, int->int, float->float
- i-j → difference , int->int, float->float
- i*j → product , int->int, float->float
- i/j → division, result is always float

- i%j → the modular operator, it gives remainder when I is divided by j
- i **j → I to the power of j

Binding variables and values

- equal sign is an assignment of a value to a variable name

```
pi = 3.141592
pi_approx = 22/7
```

- value stored in computer memory
- an assignment binds name to value
- retrieve value associated with name or variable by invoking the name, by typing pi

Abstracting expressions

- Why give names to values of expressions?
  - to reuse names instead of values
  - easier to change the code later

```
pi = 3.141592
radious = 2.2
area = pi*(radious**2)
```

To Do

When you go back home

- Make sure you read the text and understand the meaning
- Choose any three concepts and write a report on each concepts

- Install Linux using a Virtual machine or natively.
  - install following programs
    - sudo apt-get install python vi emacs spyder