Syllabus: Object Oriented Programming

Department	Math & CS	**Credits**	3	**Instructor**	Sang-Hyun Yoon	**Class Room**	5708
Subject	MC3201: OOP	**Class hrs/wk**	3	**Lab (e-mail)**		**Attendee**	
				5707 (`oop.ksa@gmail.com`)			

1 Course Description

Course Objectives

On completion of this course, you'll hopefully be able to:

- Use the Python programming language
  - Programming is needed in all areas of science and engineering, although very different languages are used.
- Think computationally
  - Computational thinking influences how you go about solving a problem
  - 50 years ago, the solution to a problem in mathematics or engineering was often a formula. Today, it is usually an algorithm.

Contents

- Python programming language
  - variables/expressions, dynamic typing
  - conditionals, loops, functions
  - lists, tuples, sets, dictionaries, strings
  - high-order functions
  - recursion
  - user-defined objects, operator overloading
  - encapsulation, inheritance
  - memory model for objects
- Problem solving using Python
  - simple robots
  - scientific computation (e.g. Gaussian elimination, N-body simulation)
  - image processing, recursive graphics
  - recursive puzzles (e.g. N-Queen, Sudoku, minesweeper)
  - winning strategy for simple combinatorial games (e.g. Tic-Tac-Toe)
  - two-player games (e.g. othello)

Prerequisites

- Programming skill acquired through MC1201/1202

2 Text & References

Text: "*The Practice of Computing using Python*", W. Punch, R. Richard

3 Grading

- Grading table

Activities	**Percentages**
Midterm	20%
Final exam	40%
Presentation/Assignments	30%
Attitude/Attendance	10%

- Absolute evaluation
- Programming assignment for each class
- Miderm exam: practical test only (3 hrs)
- Final exam: written test (0.5 hrs) & practical test (3.5 hrs)
- Late-work policy: -30%/day
- Optional bonus problems

4 Students' Presentations

- Each lecture consists of students' presentation (in English).
  - The primary goal of EC is to improve students' English.
- Each student is expected to submit a preference list of lecture units.
- Matching will be established by a variant of stable matching algorithm.
- Make your own slides or use the provided slides as it is.
- Students' lecture is evaluated by other students.

5 Lecture Schedule

0 Administrivia

- Course placement information
- Topics/Plan
- Tool installation

1. CS1/CS2 review
   - Dynamic typing
   - Variables/Expression
   - Conditionals (`if-else`)
   - Functions
   - Loops (`for, while, break/continue`)
   - Lists
   - User-defined objects
   - Problems: next permutation, square division, paper folding, pile of cubes, etc.

2. Recursion
   - Inductive definition of functions
   - Ex: Towers of Hanoi, binary search, exponentiation
   - Winning strategy of combinatorial games
   - Problems: powerset, permutation, counting, N-Queen, Sudoku, simplified Nim game, cyclic coin game, etc.

3. High-order functions
   - Operations as functions, types as sets, Functions as objects
   - High-order functions in Python
   - Anonymous functions (`lambda`)
   - Scope of names (local vs. global)
   - Default parameters, named parameters
   - Variable-length arguments
   - Problems: derivatives, integrate, curve length, surface area, Newton-Raphson method, Euler's method, etc.

4. Lists
   - Creating lists (list comprehensions)
   - `range` keyword
   - Traversal
   - Aliasing vs. copying
   - Shallow vs. deep equality

- Slicing
- As parameters & return values
- Call-by-reference vs. call-by-value
- Built-in operations/functions for lists
- Sorting under user-defined order relations

5. Built-in data structures
   - Tuples
     - As immutable lists
     - Creating tuples
     - Tuple assignment (across functions, elements of `for` loops)
     - Equality
     - Built-in operations
     - Multiple assignment (`for` loop, return value)
     - Conversion between lists/tuples
   - Sets
     - Built-in methods/operations
     - Immutability of elements in sets
     - Deep equality
     - Aliasing/copying
     - `for` loops with sets
     - Conversion between sets/lists/tuples
   - Strings
     - As unordered lists
     - As tuples of characters
     - Built-in methods
     - Formatting for `print`
   - Files
     - Reading strings from a file
     - Removing "\n" from input strings
     - `for` loops with a file object
     - Write strings to a file
     - Built-in methods
     - Another way to read from a file
   - Dictionaries
     - As a set of pairs of key/value
     - Built-in operations/methods
     - `for` loops with dictionaries
     - Immutability of keys
     - Deep equality
     - Aliasing/copying
     - Recursion with memoization ($\equiv$ dynamic programming)

6. User-defined objects I
   - Constructors, `self`

- As parameters & return values
- Object methods as pure functions vs. modifiers
- Complicated types with objects
- Operator overloading & deep equality
- Overloadable built-in operators
- Reflected arithmetic operations, comparison operations, `class` as function
- More on Wing IDE (Debugging, Help system)
- Problems: overloadable operations for rational numbers, operations for matrices over rational numbers

Midterm Exam

7. Memory layout for lists/objects
   - Memory model for lists (of lists)
   - Alias vs. shallow copy vs. deep copy of lists (of lists)
   - Lists as function inputs (call-by-value vs. call-by-reference)
   - Lists as function outputs (modifier vs. pure functions)
   - Shallow vs. deep equality for lists (of lists)
   - Memory model for user-defined objects
   - Problems: Gaussian elimination, planar figures of Rubik's cube

8. Image processing (`cs1media`)
   - negative, lighter, black-and-white (graystone/real-bw)
   - Mirror, (automatic) cropping, rotation, pixelizing vs. blurring
   - Edge detection, merging, steganography
   - Problems: sepia, reflection, blend, chromakey, posterize, printerize, flood fill

9. Graphics (`cs1graphics`)
   - Canvas, drawable object
   - Operations on drawable objects
   - Layers, animation
   - Recursive graphics: fractal

10. User-defined objects III
    - Class variables/methods, static methods
    - N-body simulation
    - Problems: 2D elastic collision, overloadable operations for vectors, N-body moving under gravitational force, Brownian motion

11. User-defined objects III
    - Encapsulation, private variables/methods
    - Inheritance, multiple inheritance, `super`
    - Tic-Tac-Toe game
    - Problems: inherited classes for line, ling segment, and half-lines, winning strategy for Tic-Tac-Toe

12. Recursive puzzles
    - N-Queen with graphics
    - Sudoku with graphics

13. Recursive games
    - Minesweeper
    - Othello

Final Exam

6 Lecture Schedule (Tabular)

Lec #	Topics	Subtopics
1	CS1/CS2 review	
2	Recursion	combinatorial games
3	High-Order Functions	scope, default param.
4	Lists	sort
5	Built-In Data Structures	set/dictionary/tuple/..
6	User-Defined Objects I	debugger
	Midterm Exam	
7	Memory Layout for Objects	gaussian elimination
8	Image Processing	
9	Graphics	
10	User-Defined Objects II	N-body simulation
11	User-Defined Objects III	tic-tac-toe
12	Recursive Puzzles	N-queen, sudoku
13	Recursive Games	minesweeper, othello
	Final Exam	

Lecture Units for Students' Presentation

- score = max(20, $\sum$ credits $\times$ evaluation (from other students))
  - If you selected two units with credits 12 and 15 and the evaluation points is 45/100 and 95/100, respectively, then the score is max(20, 12*0.45+15*0.95) = 19.65
- Each team consists of 1$\sim$3 students.

Class No: Student ID: Name :

Class No: Student ID: Name :

Class No: Student ID: Name :

No.	Topics	Credits	Preference
L1	CS1/CS2 Review		N/A
L2	Recursion	18	
L3	High-Order Functions	14	
L4	Lists	13	
L5	Built-In Data Structures	12	
L6	User-Defined Objects I	10	
	Midterm Exam		
L7	Memory Layout for Objects	18	TBD
L8	Image Processing	12	TBD
L9	Graphics	14	TBD
L10	User-Defined Objects II	10	TBD
L11	User-Defined Objects III	10	TBD
L12	Recursive Puzzles		N/A
L13	Recursive Games		N/A

- Matching will be established by a variant of stable matching algorithm.
- If some unit would not be selected by any students, it will be randomly assigned to a student.
- Mark as many units as possible to avoid topic that you don't want.

Example:

No.	Topics	Credits	Preference
L2	Recursion	18	**1**
L3	High-Order Functions	14	**4**
L4	Lists	13	**2**
L5	Built-In Data Structures	12	**5**
L6	User-Defined Objects I	10	**3**