ITEC201006

Introduction to Computer Science & Engineering

Lecture 8: Objected-Oriented Design

Jeonghun Park

KNU KYUNGPOOK NATIONAL UNIVERSITY

Object-Oriented Design

객체 지향 디자인

- Object-oriented Design 흔히 사용하는 function을 미리 만들어두고 가져다 쓰는것
  - A problem-solving methodology that produces a solution to a problem in terms of self-contained entities called *objects*

- Object Input, output 이 정해진 Function
  - A thing or entity that makes sense within the context of the problem
  - For example, a *student*, a *car*, *time*, *date*

jeonghun.park@knu.ac.kr

Object-Oriented Design

- Problems are solved by
  - isolating the objects in a problem, ① 문제 내 객체 찾기
  - determining their properties and actions (responsibilities), and ② 객체마다 역할을 줌
  - letting the objects collaborate to solve a problem ③ 역할을 부여받은 객체끼리 조합

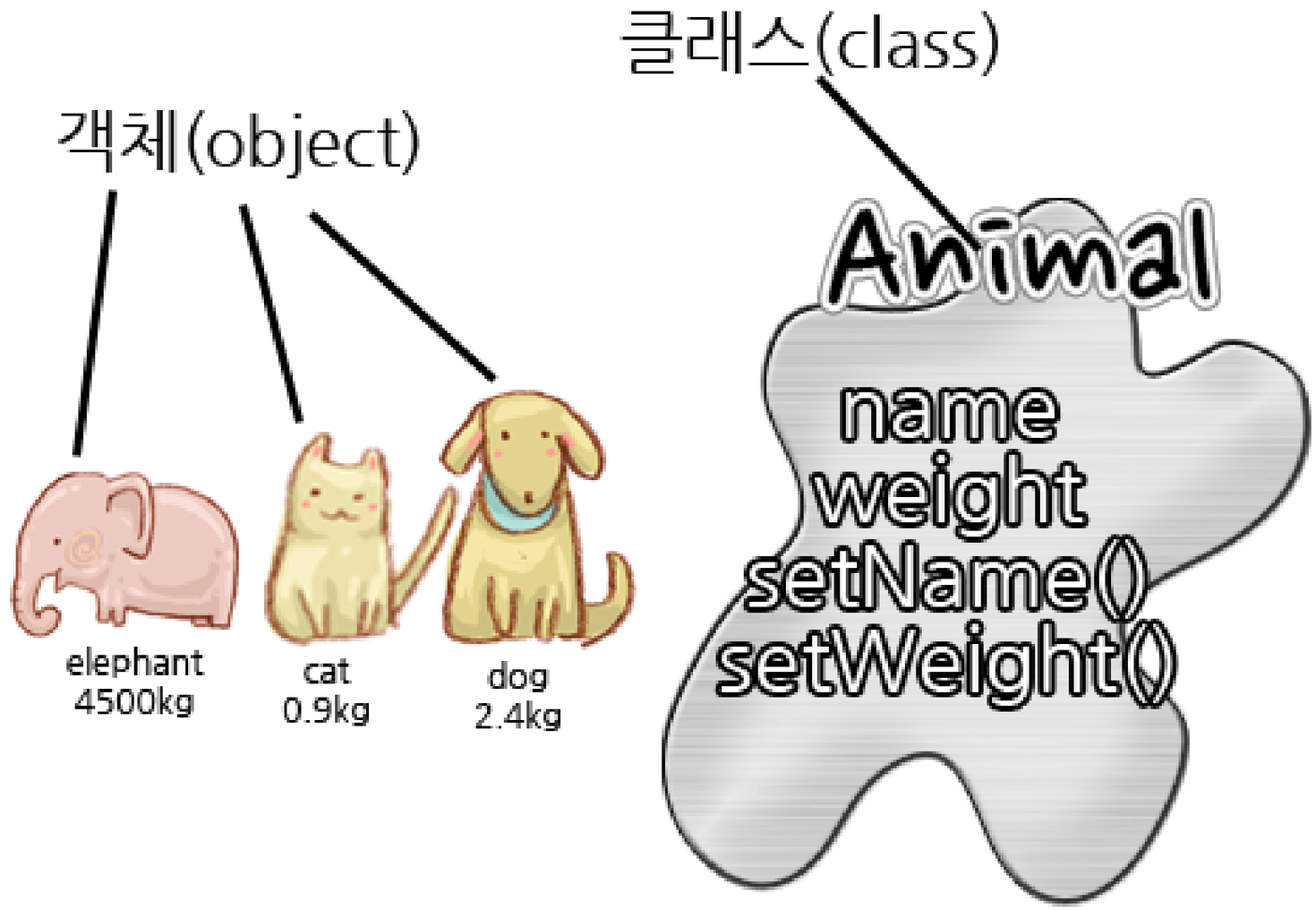
jeonghun.park@knu.ac.kr

Object-Oriented Design

- Class (or object class) 유사한 object 의 집합
  - A description of a *group* of similar objects
- Object (instance of a class)
  - A concrete example of the class
- Classes contain fields that represent the
  properties (name, eye color) and
  behaviors (responsibilities) (shop, cook) of the class

jeonghun.park@knu.ac.kr

Object-Oriented Design

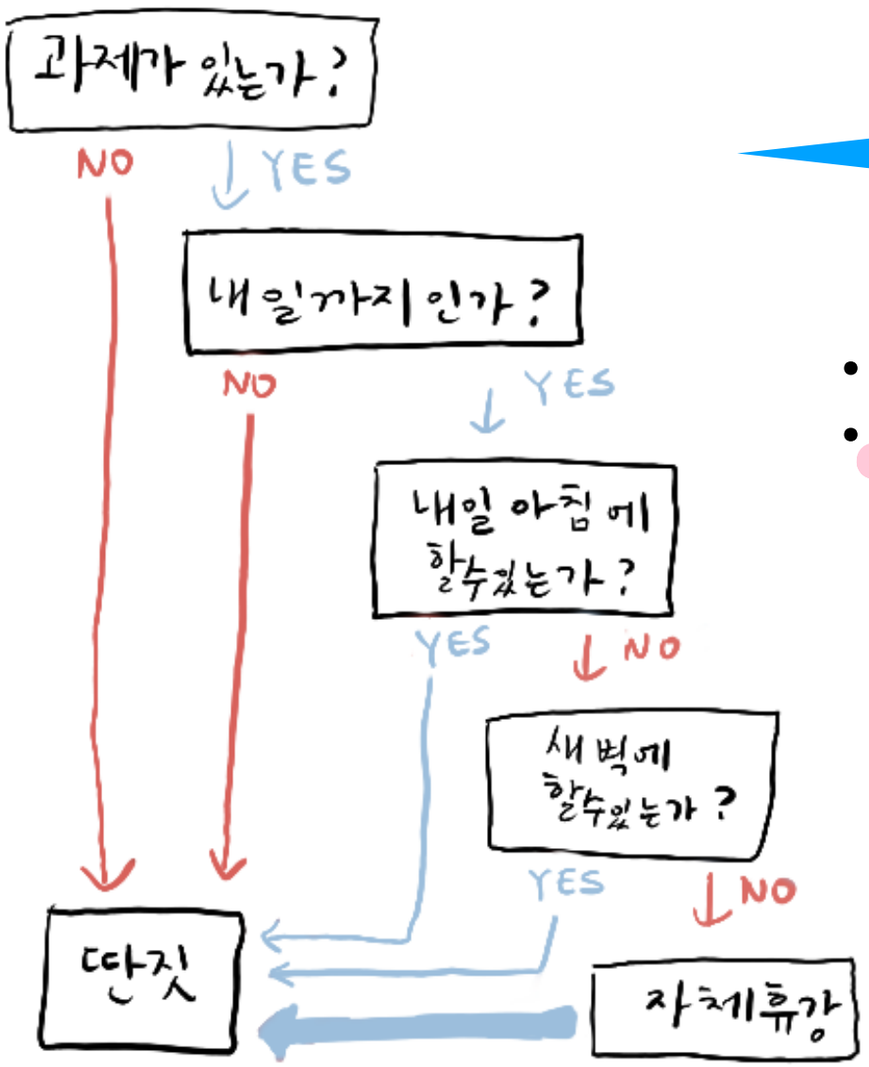


jeonghun.park@knu.ac.kr

What is Not Object-Oriented Design?

〈절차지향〉

과제 알고리즘



ample of [Procedural Design]

- Basically a collection of commands
- Hardly reusable

이 프로그램 내에서만 유용.

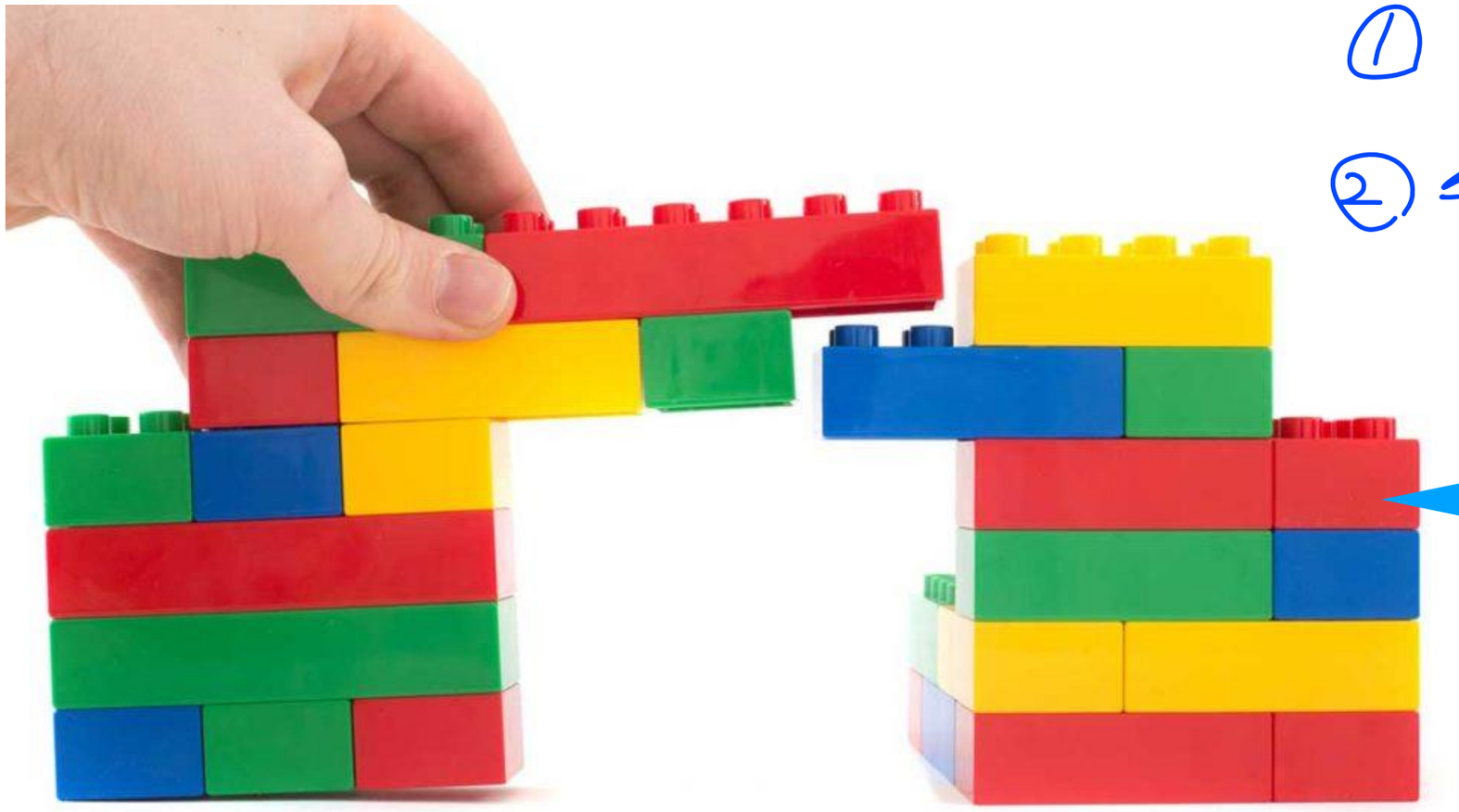
jeonghun.park@knu.ac.kr

Why OOD?

Object-Oriented Design

① 재사용이 쉽다.

② 수정/이용이 유연하다.



Object

- Basically a collection of data processing functions
- Easily reusable

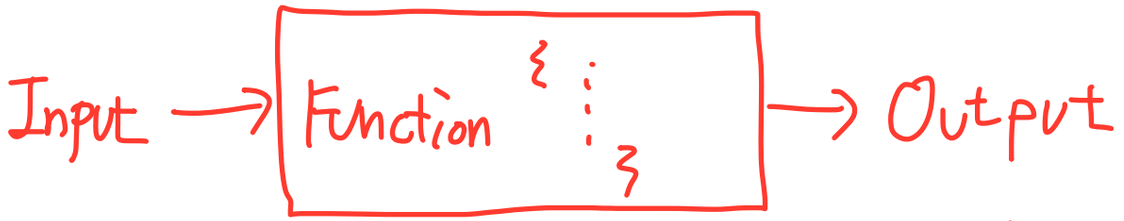
jeonghun.park@knu.ac.kr

Fundamentals

- Objects
  - Basically, OOD is a collection of objects, which have a particular purpose
  - Each object has 1) data, 2) a collection of behaviors, and 3) identity
- Classes
  - Blueprint of objects
  - Similar to [Type and Variable]
  - For example, int i, int j etc..

jeonghun.park@knu.ac.kr

Fundamentals



- Encapsulation * function 이 어떻게 코딩되었는 몰라도 쓰는방법만 알면 적절한결과를 나타냄.
  - Bundling of the the variables and functions
  - Purpose: Reuse the code without any modification
- Information hiding function은 캡슐화처럼 쓰게 해주지만 코드는 못보게함
  - Some particular modules can be hidden by minimizing exposure to outside of the modules
  - Can be considered as a byproduct of encapsulation

jeonghun.park@knu.ac.kr

Fundamentals

- Inheritance 상속

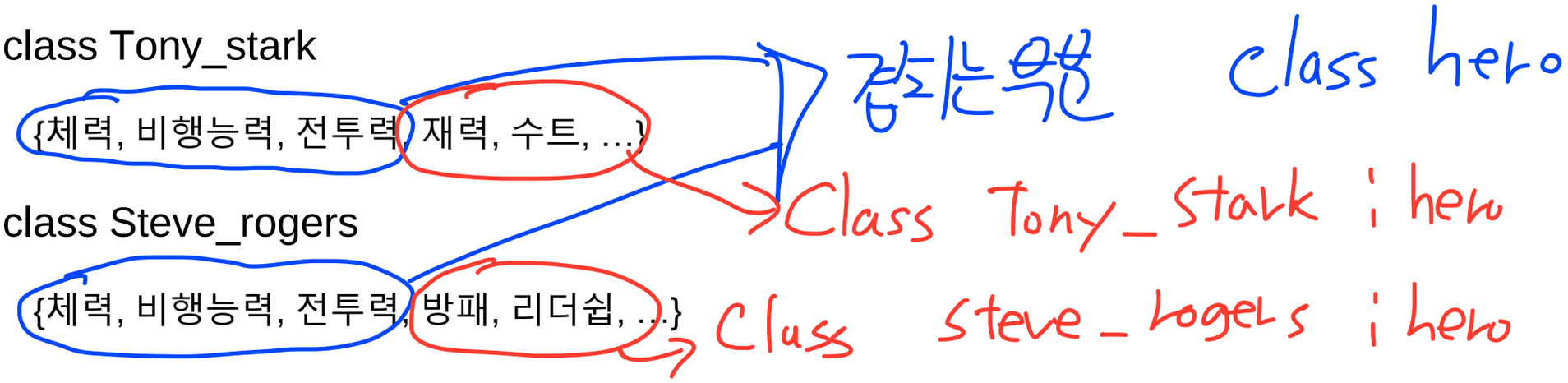
  - If we want to design Avengers game, we might have:

    - class Tony_stark
      - {체력, 비행능력, 전투력, 재력, 수트, …}
    - class Steve_rogers
      - {체력, 비행능력, 전투력, 방패, 리더쉽, …}

  - But those two classes have similar features

  - So we rather have:

    - class hero
      - {체력, 비행능력, 전투력}
    - class Tony_stark : public: hero
      - {재력, 수트, …}



es flexible reuse of the codes

jeonghun.park@knu.ac.kr

Fundamentals

- Polymorphism 다형성.
  - One particular function can be interpreted as different meaning depending on situations

특정 object가 상황에 따라 다른 역할을 할수 있다.

jeonghun.park@knu.ac.kr

Pros/Cons

- Pros
  - High level coding is possible
  - A code becomes flexible
    - Reuse is possible
- Cons
  - Levels of difficulties increase
  - Overuse of public variable (particularly in inheritance) will make the code very complicated
  - Education is difficult

jeonghun.park@knu.ac.kr

Design Way

- Similar to what we've learned in the computer science problem
  - Never reinvent the wheel!
- Let's practice with the project problem
  - Exploitation vs Exploration in Huffman encoding
  - In fact, however, we do not need OOD in the level of HW or project

jeonghun.park@knu.ac.kr

Compilers

- High-level language 고급언어
  - A language that provides a richer (more English-like) set of instructions

- Compiler
  - A program that translates a high-level language program into machine code

C 언어 (High level Language) → 컴파일러 (compiler) → 프로그램 (Binary) machine code

jeonghun.park@knu.ac.kr

Compilers

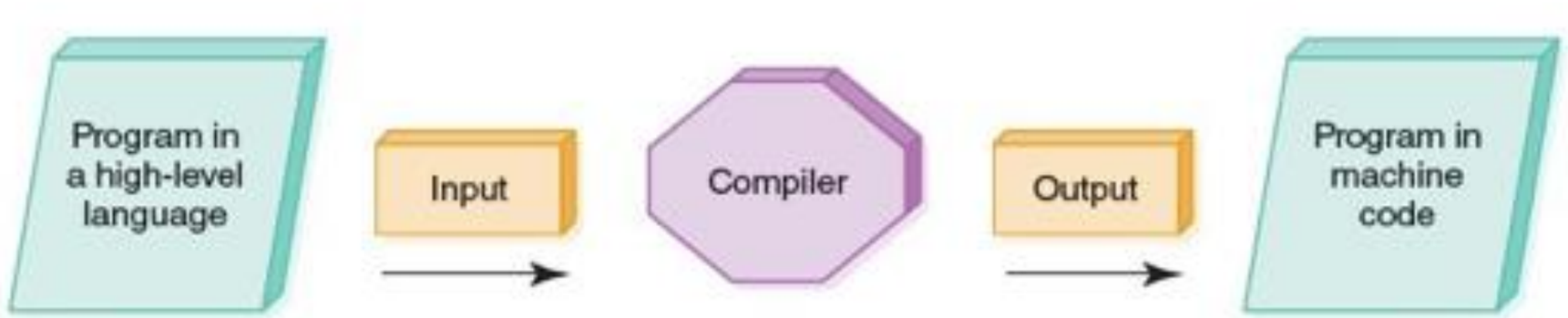


FIGURE 9.2 Compilation process

jeonghun.park@knu.ac.kr

Interpreters

- Interpreter 줄마다 바로실행 (Matlab)
  - A translating program that translates and executes the statements in sequence
  - Assembler or compiler produce machine code as output, which is then executed in a separate step
  - An interpreter translates a statement and then immediately executes the statement
  - Interpreters can be viewed as *simulators*

jeonghun.park@knu.ac.kr

Functionality of High-Level Languages

- Sequence
  - Executing statements in sequence until an instruction is encountered that changes this sequencing
- Selection
  - Deciding which action to take
- Iteration (looping)
  - Repeating an action

jeonghun.park@knu.ac.kr

Data Types

- Boolean expression
  - A sequence of identifiers, separated by compatible operators, that evaluates to *true* or *false*
  - A Boolean expression can be
    - A Boolean variable
    - An arithmetic expression followed by a relational operator followed by an arithmetic expression
    - A Boolean expression followed by a Boolean operator followed by a Boolean expression

jeonghun.park@knu.ac.kr

Data Types

- Integer numbers
- Real numbers
- Characters (ASCII)
- Boolean values
- Strings
- Which types are in the project problem?

jeonghun.park@knu.ac.kr

Declarations

선언 (변수선언)

- Declaration
  - A statement that associates an **identifier** with a **variable**, an **action**, or some other **entity** within the language that can be given a name; the programmer can refer to that item by name
- Reserved word <예약된 단어>
  - A word in a language that has special meaning
- Case-sensitive
  - Uppercase and lowercase letters are considered the same (Not in Matlab)

소,대문자 구분한 Matlab 이전 다름.

jeonghun.park@knu.ac.kr

Declaration Example

Language	Variable Declaration
Python	None required

VB .NET	`Dim sum As Single = 0.0F ' set up word with 0 as contents`
	`Dim num1 As Integer ' set up a two byte block for num1`
	`Dim num2 As Integer ' set up a two byte block for num2`
	`Dim num3 As Integer ' set up a two byte block for num3`
	`...`
	`Num1 = 1`

C++/Java	`float sum = 0.0; // set up word with 0 as contents`
	`int num1; // set up a two byte block for num1`
	`int num2; // set up a two byte block for num2`
	`int num3; // set up a two byte block for num3`
	`...`
	`Num1 = 1;`

ot required in MATLAB!

jeonghun.park@knu.ac.kr

Assignment statement

- Assignment statement
  - An action statement (not a declaration) that says to evaluate the expression on the right-hand side of the symbol and store that value into the place named on the left-hand side
- Named constant
  - A location in memory, referenced by an identifier, that contains a data value that cannot be changed

In MATLAB, when we assign, it is automatically declared!

jeonghun.park@knu.ac.kr

Input/Output Structures

- Pseudocode algorithms used the expressions
  - *Read* or *Get* and *Write* or *Print*
- High-level languages view input data as a stream of characters divided into lines
- Key to the processing
  - The data type determines how characters are to be converted to a bit pattern (input) and how a bit pattern is to be converted to characters (output)

jeonghun.park@knu.ac.kr

Let's Practice

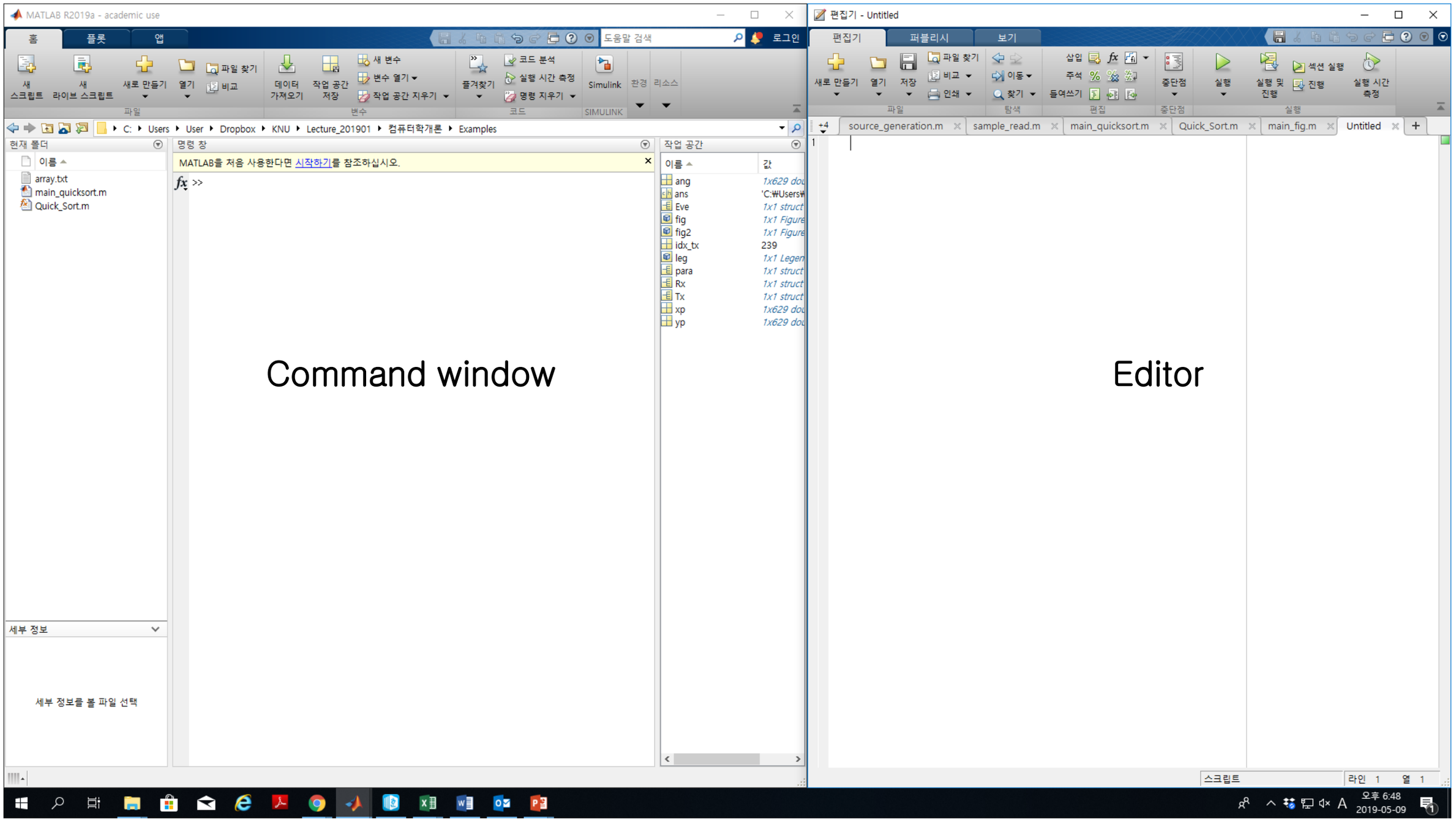
- Let's practice using a MATLAB example code
  - Quicksort algorithm example

- MATLAB is a high-level programming language useful for engineering
  - Easy to learn, easy to use
  - Design degrees of freedom might be low 자유도는 낮다
    - So it might not be appropriate for complicated programming

복잡한 프로그래밍은 안될수도 있다.

jeonghun.park@knu.ac.kr

MATLAB Structure



MATLAB Example Code

```
clc; close all; clear;

x = rand(10,1);
y = Quick_Sort(x);
t = 1:length(x);

figure();
bar(t, x, 'b');
grid on;

figure();
bar(t, y, 'r');
grid on;

f_id = fopen('array.txt', 'w');

fprintf(f_id, 'Our unsorted array is :₩n');
fprintf(f_id, '%f₩n', x);

fprintf(f_id, 'Our sorted array is : ₩n');
fprintf(f_id, '%f₩n', y);
```

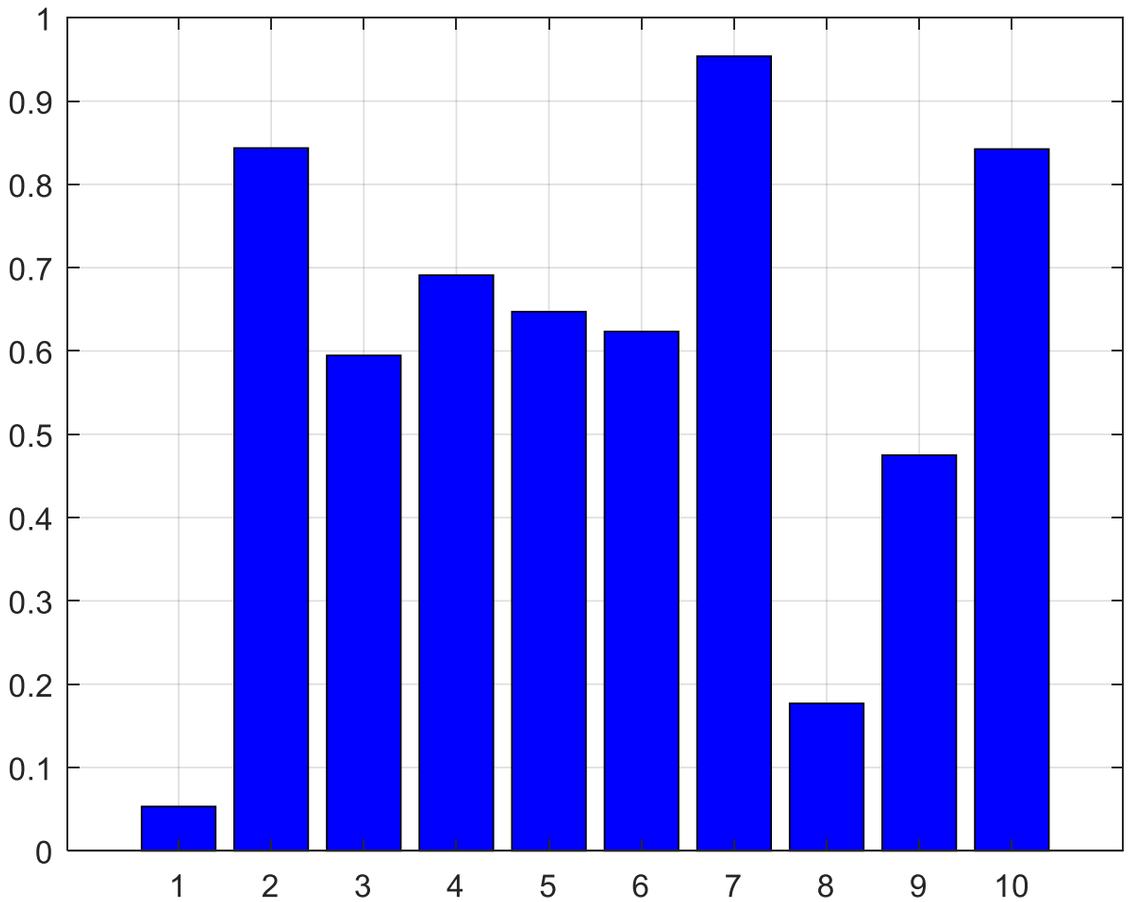
Quicksort algorithm

MATLAB Example Code

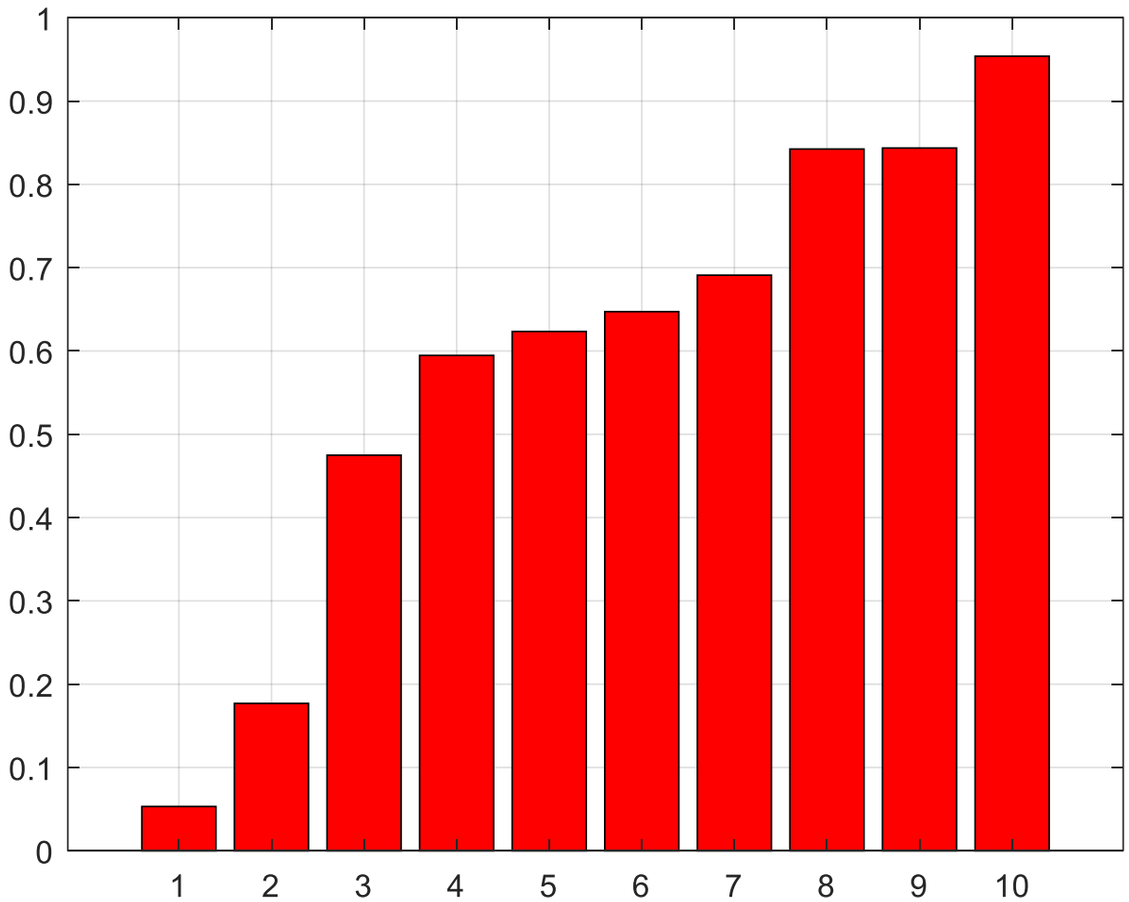
```
function y = Quick_Sort(x)

n=length(x);

if n <= 2
    y = x;
    return;
end
x1 = [];
x2 = [];
for i = 1:n-1
    if x(i) <= x(n)
        x1 = [x1 x(i)];
    else
        x2 = [x2 x(i)];
    end
end

x1
x2

y = [Quick_Sort(x1) x(n) Quick_Sort(x2)];

end
```

Observe how IF and WHILE statement are used to construct this program

MATLAB Example Code



x (unsorted array)



y (sorted array)

jeonghun.park@knu.ac.kr

MATLAB Example Code

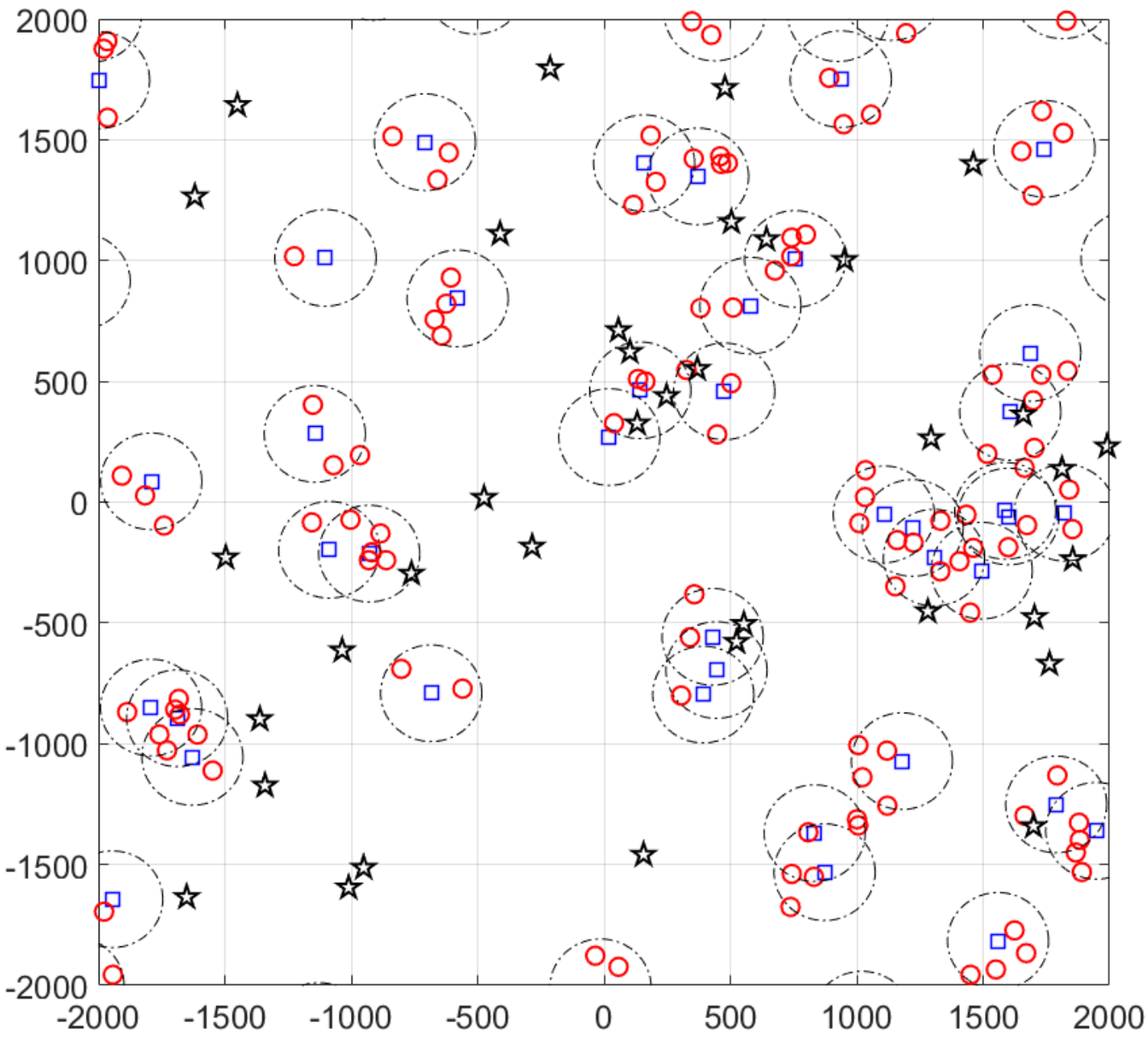
```
clc; close all; clear;

para.lambdaTx = 0.1*10^(-4)/pi; % BS Density
para.lambdaEve = 5*10^(-6)/pi; % BS Density
% para.lambdaUser = 1*10^(-4)/pi; % User Density
para.L = 5000;
para.Area_total = pi*para.L^2; % Total area (for wrap-around)
para.Rd = 200;
% para.b = 4; % pathloss exponent
% para.iterNum = 1000; % iteration number


%% Tx Drop

Tx.Number = poissrnd(para.Area_total*para.lambdaTx);

Tx.Radius = sqrt(unifrnd(0,para.L^2,Tx.Number,1));
Tx.Theta = unifrnd(0,2*pi,Tx.Number,1);

Tx.PosX = Tx.Radius.*cos(Tx.Theta);
Tx.PosY = Tx.Radius.*sin(Tx.Theta);

%% Rx Drop

for idx_tx = 1:Tx.Number

    Rx.Number{idx_tx} = randi(4);

    Rx.Radius{idx_tx} = sqrt(unifrnd(0,para.Rd^2,Rx.Number{idx_tx},1));
    Rx.Theta{idx_tx} = unifrnd(0,2*pi,Rx.Number{idx_tx},1);

    Rx.PosX{idx_tx} = Tx.PosX(idx_tx) + Rx.Radius{idx_tx}.*cos(Rx.Theta{idx_tx});
    Rx.PosY{idx_tx} = Tx.PosY(idx_tx) + Rx.Radius{idx_tx}.*sin(Rx.Theta{idx_tx});

end
```

Wireless network model construction

MATLAB Example Code

```
Eve.Number = poissrnd(para.Area_total*para.lambdaEve);

Eve.Radius = sqrt(unifrnd(0,para.L^2,Eve.Number,1));
Eve.Theta = unifrnd(0,2*pi,Eve.Number,1);

Eve.PosX = Eve.Radius.*cos(Eve.Theta);
Eve.PosY = Eve.Radius.*sin(Eve.Theta);


fig = figure();
% plot(Tx.PosX, Tx.PosY, 'b^','linewidth', 1.5, 'markersize', 10, 'MarkerFaceColor',[0 0 1]);
plot(Tx.PosX, Tx.PosY, 'bs','linewidth', 1.5, 'markersize', 10);
grid on;
hold on;
for idx_tx = 1:Tx.Number

    %     plot(Rx.PosX{idx_tx}, Rx.PosY{idx_tx}, 'ro','linewidth', 1.5, 'markersize', 10, 'MarkerFaceColor',[1 0 0]);
    plot(Rx.PosX{idx_tx}, Rx.PosY{idx_tx}, 'ro','linewidth', 1.5, 'markersize', 10);

    ang=0:0.01:2*pi;
    xp = para.Rd*cos(ang);
    yp = para.Rd*sin(ang);

    plot(Tx.PosX(idx_tx) + xp, Tx.PosY(idx_tx) + yp, '-.k','linewidth', 0.1);
end
plot(Eve.PosX, Eve.PosY, 'kp','linewidth', 1.5, 'markersize', 13);

xlim([-2000, 2000])
ylim([-2000, 2000])
% set(fig1, {'MarkerFaceColor'}, get(fig1,'Color'));

set(gca,'fontsize',18);
```

MATLAB Example Code



MATLAB Example Code

```
clc; close all; clear;

Accum_prob = cumsum(Prob);

%% Sample number is 1e5
sample_num = 1e5;

for idx_sample = 1:sample_num

    random_seed = rand;
    for idx_source = 1:9
        if idx_source == 1
            if (random_seed <= Accum_prob(idx_source))
                Sample(idx_sample) = char(idx_source + 96);
            end
        else
            if (random_seed > Accum_prob(idx_source-1)) && (random_seed <= Accum_prob(idx_source))
                Sample(idx_sample) = char(idx_source + 96);
            end
        end
    end
end

%% Sample write
f_id = fopen('source_JPARK2019_vfinal.txt', 'w');
for idx_sample = 1:sample_num
    fprintf(f_id, '%2s₩n', Sample(idx_sample));
end
```

Understand how the sample is made

What is this code?

jeonghun.park@knu.ac.kr

MATLAB Example Code

```
clc; close all; clear;

f_id = fopen('source_JPARK2019_vfinal.txt', 'r');
format_read = '%2s';

Sample_r = fscanf(f_id,format_read);

OurSource = [];
SourceCount = [];

sample_length = length(Sample_r);

for idx = 1:sample_length
    if ismember(Sample_r(idx), OurSource)
        sample_idx = find(Sample_r(idx) == OurSource);
    else
        OurSource = [OurSource, Sample_r(idx)];
    end
end

for idx = 1:length(OurSource)
    SourceCount(idx) = count(Sample_r, (OurSource(idx)));
end
```

Let's study line-by-line of this code. It is possibly useful for the project!