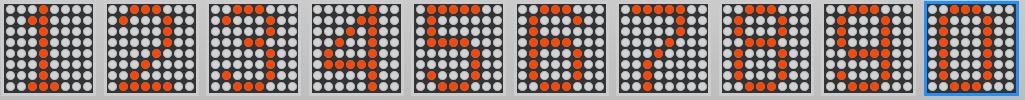
|  |
| --- |
| Computer Science Student |
| PORTFOLIO |
| Ivan Derianto |

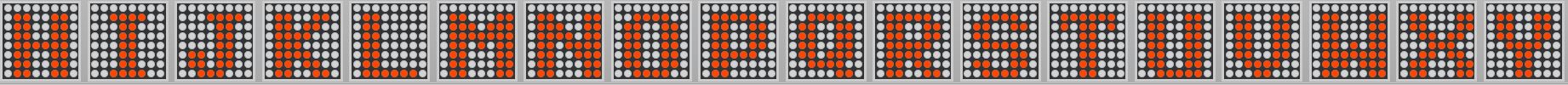
|  |
| --- |
|  |

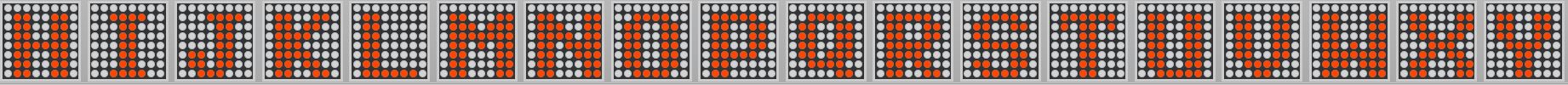
**Alphanumeric Recognition using Perceptron (Single Layer Neural Network)**

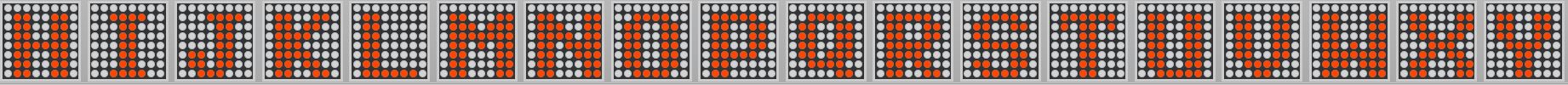
**Programming Language : C**

AlphaNumeric consists 10 numbers (0-9) and 26 alphabets (A-Z). We are using 8x8 matrix to create the alphanumeric in binary mode. So, we will have 64 inputs (8x8) in total + 1 Bias (the value of bias is 1). We will have 36 results (0-9, A-Z) which means we will train the neural network to do 36 patterns repetition until we find the result we want.



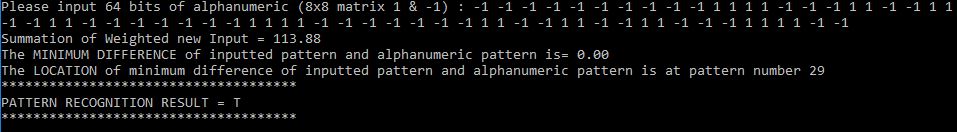






I use these patterns as a guide to create the binary file. Red dot represents ‘1’ value and the white dots represents ‘-1’ value. The binary number of each pattern is stored in alphanumeric.txt that will be called by function in the C program. The first 10 pattern is numeric (0-9) and the rest of them is alphabet (A-Z). The last digit of each pattern is set to 1 (destination result value). At the final row 999 means the end of the file.

The result will print every set Pattern with 64 input values + bias, and 65 RANDOM weight values + weight of bias. Input[65] and weight[65] are for bias. The training process will take several steps using well-known Perceptron algorithm, including weight update to produce the final weight which will be used to recognize the new inputted simple alphanumeric pattern.



**Face Recognition using Hopfield Network Algorithm**

**Programming Language : C++**

For this program, I’m using 3 20x20 pixel samples. The images will be blurred badly because 20x20 pixel is extremely small resolution.

C:\Users\Ivan\AppData\Local\Microsoft\Windows\INetCache\Content.Word\test1.jpg test1.jpg

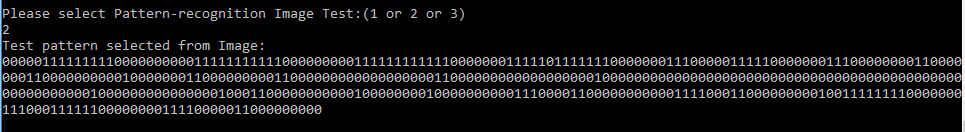
C:\Users\Ivan\AppData\Local\Microsoft\Windows\INetCache\Content.Word\test2.jpg test2.jpg

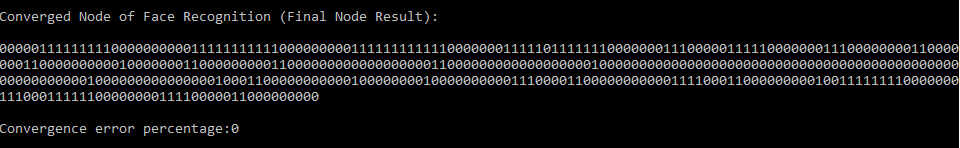
C:\Users\Ivan\AppData\Local\Microsoft\Windows\INetCache\Content.Word\test3.jpg test3.jpg

**C++ - OpenCV Library**

I use OpenCV library to perform Image Processing. In this program, we need to read images, convert them to Grayscale, then we convert the Grayscale images to Black and White images using threshold method. Value greater than 128 will set to white, value less than 128 will set to black. After converting the image, we can get binary values of each pixel; White = 0, Black = 1.

The binary pixel values of 3 BW images above is used for the training pattern. Weight value calculation, node update and other processes are done using the Hopfield Network algorithm.

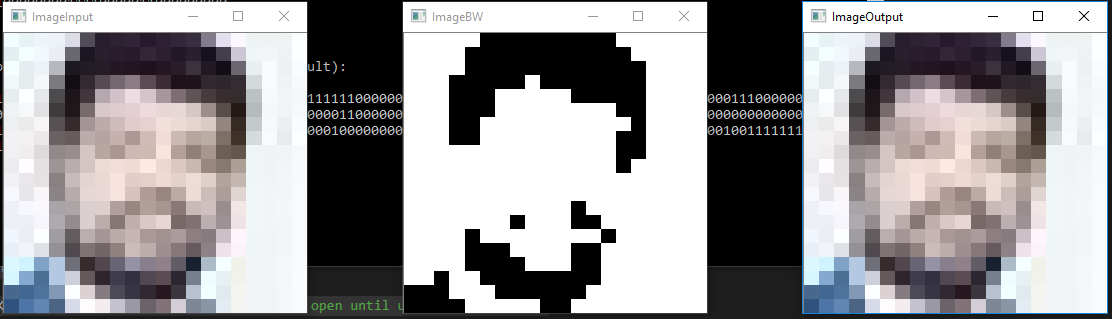




**Displaying Images for Final Result**

Original Image that has been inputted – Conversion to Grayscale then to Black and White image – Successfully recognized face result

As we can see the ImageOutput window shows the same result as the ImageInput window, which mean the face of the person is successfully recognized.



**Movie Review Sentiment Analyzer**

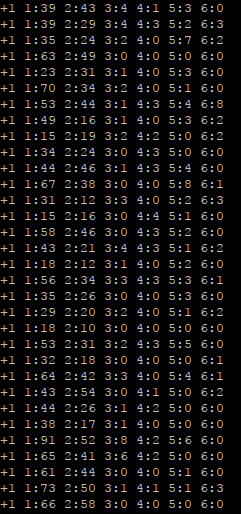
**Programming Language : Python**

The purpose of this program is to determine the best set of features (properties of a problem based on which we would like to predict results in machine learning) for automatically analyzing the sentiment or polarity (Neutral, Positive, Negative) of movie review dataset.

Features Sets:

* Set 1:
  + Total of positive words
  + Total of negative words
  + Total of positive words in the first sentence
  + Total of negative words in the first sentence
  + Total of positive words in the last sentence
  + Total of negative words in the last sentence
* Set 2: Word frequency (total occurrence of each word)
* Set 3:
  + Adjective Extraction (with restrictions)
  + Total of contrast words
  + Total of negation words
  + Total of power(exaggeration) words

**Class Classification**

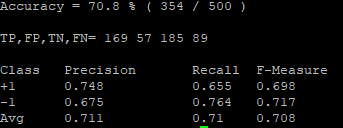
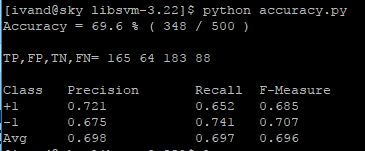
I use SVM (support vector machine) to analyze the data and perform the classification of each class. Specifically, I use LIBSVM (software for SVM classification and regression). All of the features needed are extracted using python and the results are saved in LIBSVM format.

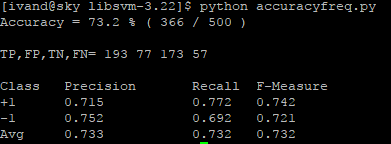
result of the first set that contains 6 features in LIBSVM format

Data training process (10-fold cross validation), model creation, prediction and classification are all done using the provided LIBSVM libraries. To choose the best performance among those 3 sets of features, precision, recall, and **f-measure** score are calculated and compared.

Set1

Set2



**Citation Purpose and Sentiment Analyzer**

Set3

**Programming Language : Python**

Similar to movie review sentiment analyzer, but in this project, I extract different sets of features which are more complex with several methods (usual extraction and CRF++). The purpose of this project to determine the best set of features for automatically analyzing the sentiment and the purpose of citing sentence in scientific paper.

Features Sets:

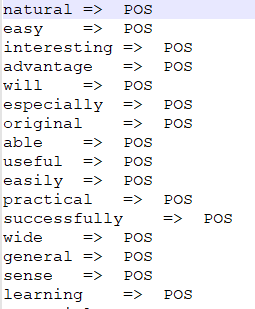
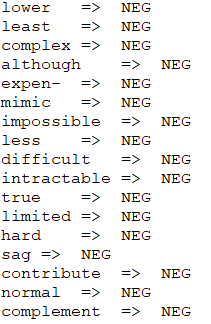
* Set 1 consisting 11 features
  + References
  + Contrast words
  + Negation words
  + Comparison words
  + Adjectives
  + Verbs
  + Adverbs
  + First Person Pronouns
  + Third Person Pronouns
  + Location of target reference
  + Positive adjective in sentence that contains target reference
  + Negative adjective in sentence that contains target reference
* Set 2: Word frequency using CRF++ (Conditional Random Field for labeling sequential data) and MPQA Subjectivity Lexicon

CRF++ is described in <https://taku910.github.io/crfpp/>

MPQA Subjectivity Lexicon can be downloaded from here [mpqa.cs.pitt.edu/lexicons/subj\_lexicon/](http://mpqa.cs.pitt.edu/lexicons/subj_lexicon/)

**CRF++**

The MPQA Subjectivity Lexicon already has its subjectivity clues. In order to make comparison I use CRF++ to create new subjectivity dictionary (like the one from MPQA) from raw dataset. First the training dataset is labeled with positive and negative tag for each word that listed in MPQA subjectivity lexicon. Then the tagged dataset must be converted into IOB format as stated in CRF++ tutorial. Training (model production) and testing are done using crf commands to produce new positive and negative clues.



To choose the best performance among those 3 sets of features, precision, recall, and **f-measure** score are calculated and compared.

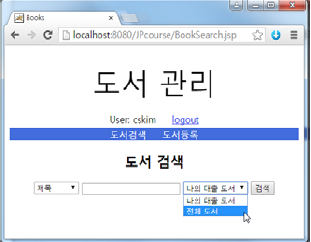
**Library Management System (Web Based)**

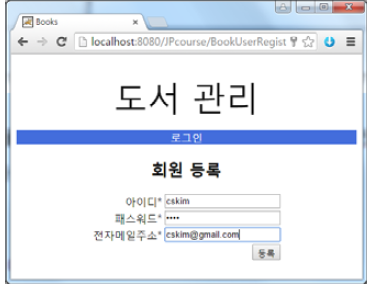
**Languages : Java, JSP, HTML, CSS**

In this project, I created a simple library management system website using JSP as the main web programming language, and MySQL for managing the database.

Several features that included in this website are:

* User registration
* User login - logout
* Book registration
* Book search engine
  + Search all
  + Search for the borrowed books by the user
  + Multiple options (book title, author, publication year, publisher)
* Borrow – Return system
  + The borrowed book by this user can’t be borrowed by another user

This project is done in Korean, so all the writing will be in Korean word.

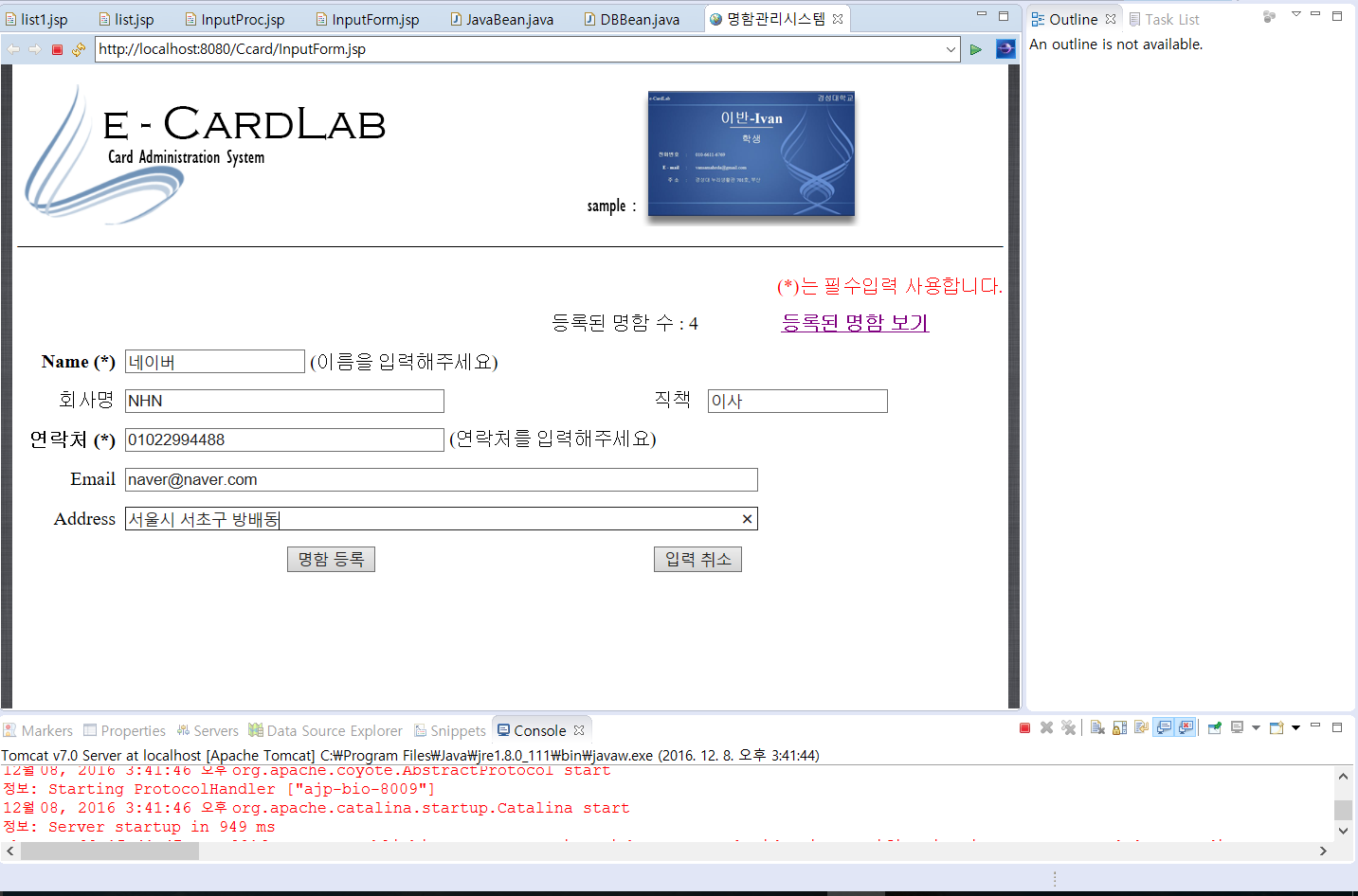
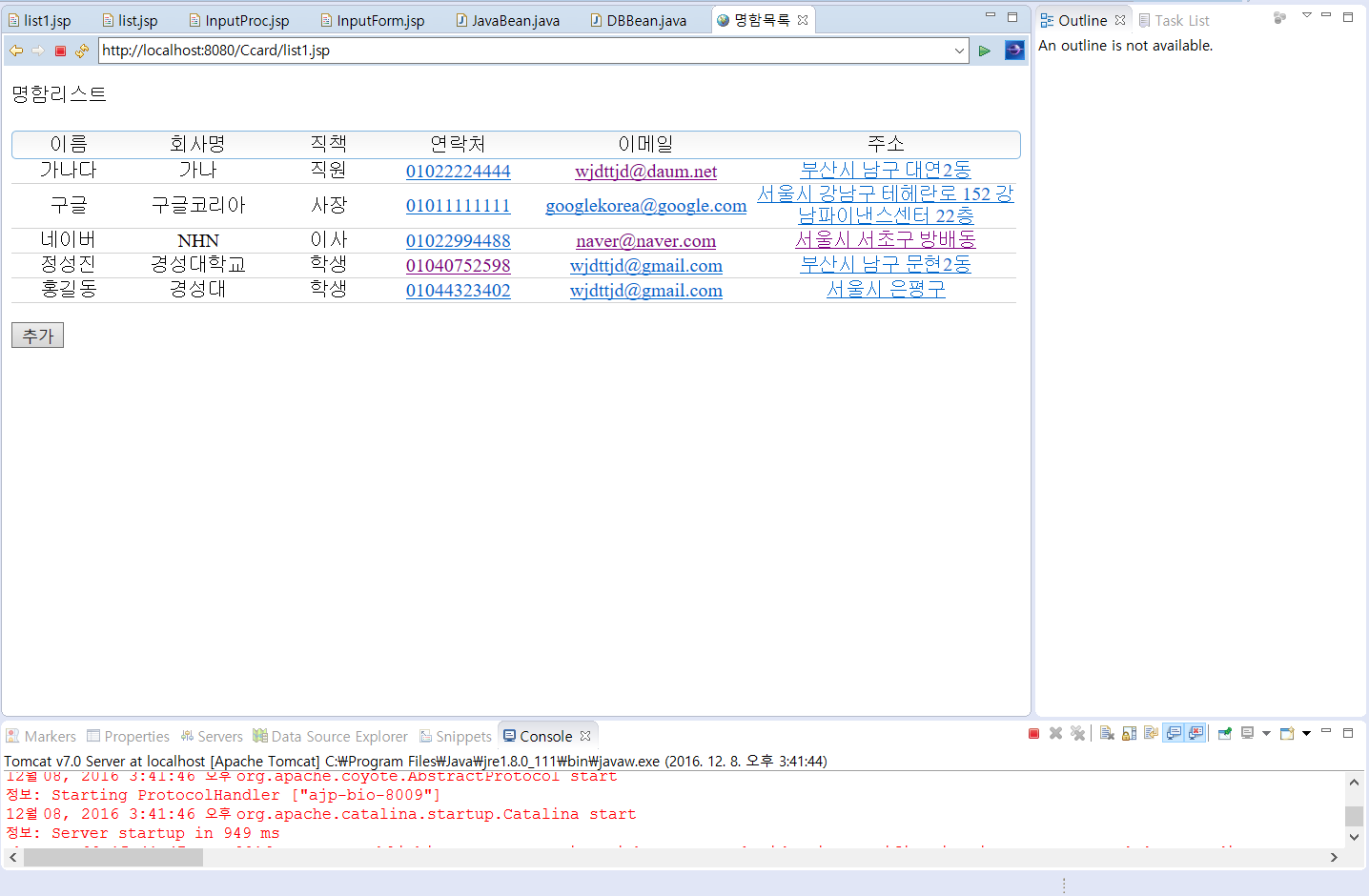




**Business Card Management System**

**Languages : Java, JSP, HTML, CSS**

This project is created using JSP as the main webpage scripting language. We can input the business card data and store all the data in database using JDBC (Java Database Connectivity) and MySQL to manage the database. This project is done both in eclipse and the web browser (target). This project also has its own mobile web version, which will make viewing on smartphone becomes easier. The output or printed data on the virtual card which includes user’s phone number, email and address could be clicked and direct you to corresponding application such as google map, Gmail, and call menu on smartphone or

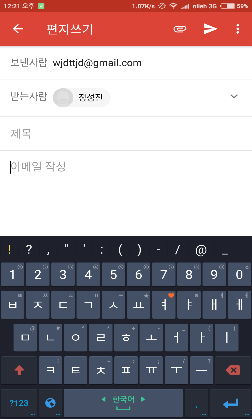
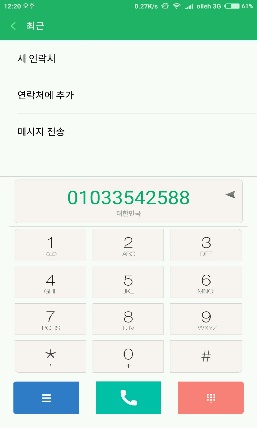
Microsoft outlook.

Smartphone browser

PC browser







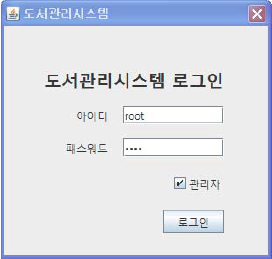
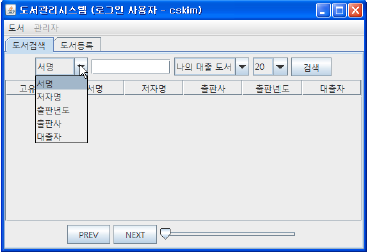
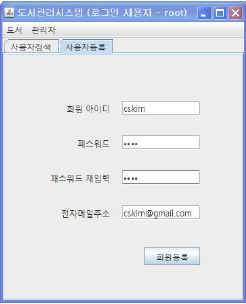
**Library Management System (Window Builder Based)**

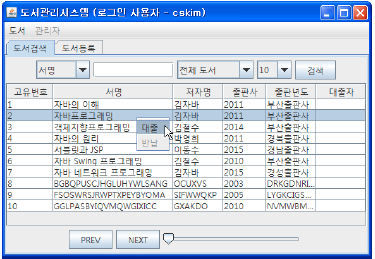
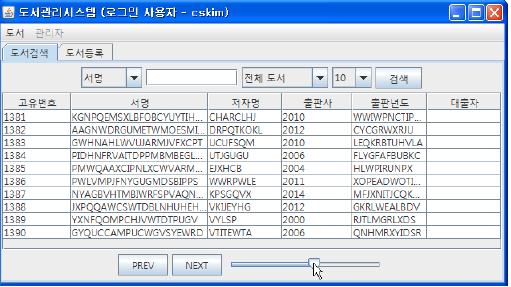
**Programming Language : Java**

WindowBuilder is composed of SWT Designer and Swing Designer and makes it very easy to create Java GUI applications. The overall project is similar to web based library management system project, with some different features and platform.

Several features that included in this website are:

* Administrator (user with certain privileges)
  + Book registration
  + User management
* User registration
* User login - logout
* Book search engine
  + Search all
  + Search for the borrowed books by the user
  + Multiple options (book title, author, publication year, publisher)
* Borrow – Return system
  + The borrowed book by this user can’t be borrowed by another user



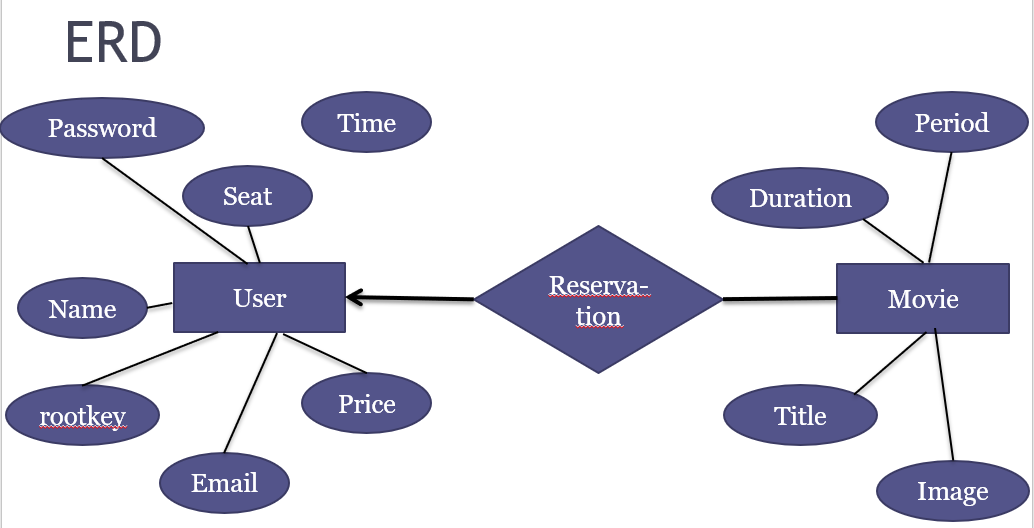
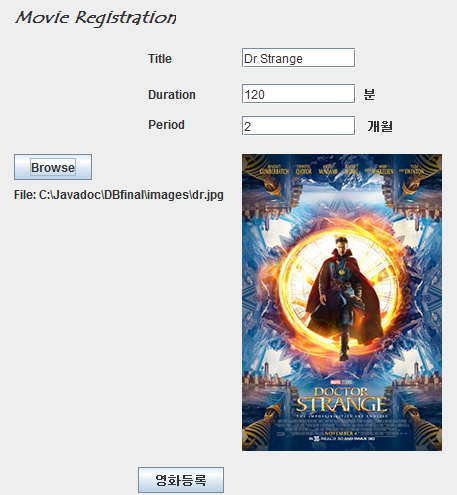


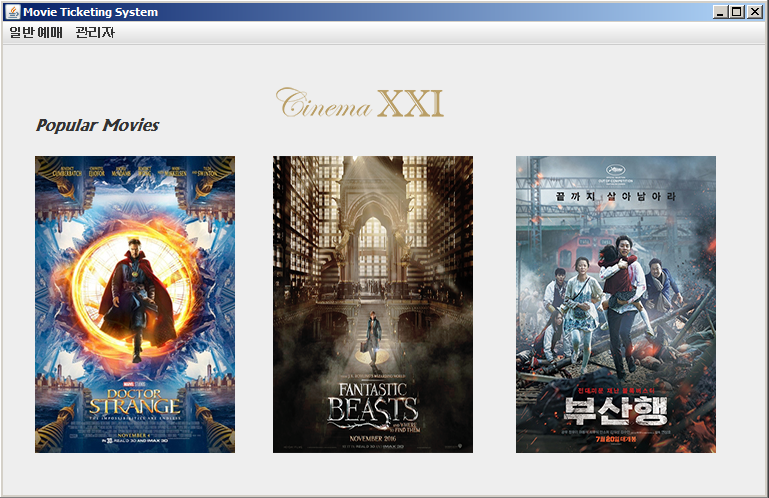
**Movie Ticketing System**

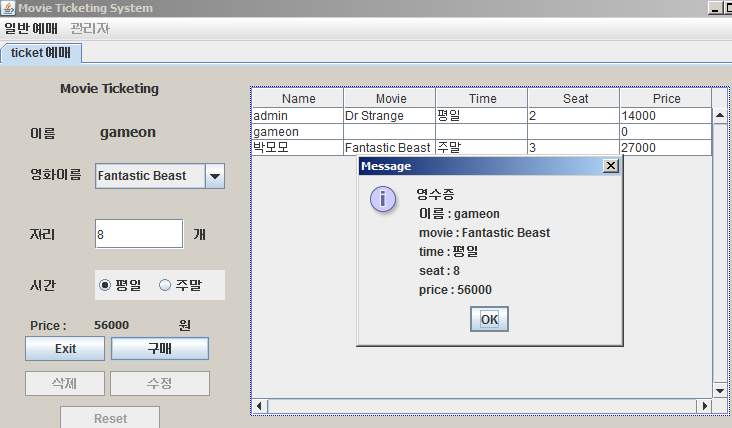
**Programming Language : Java**

This movie ticketing system is a simple Java GUI application that built using WindowBuilder. All data are stored in database and managed using MySQL. This project consists several features:

* Movie ticket reservation
* Administrator
  + Movie registration
  + User management
* User registration
* User login – logout



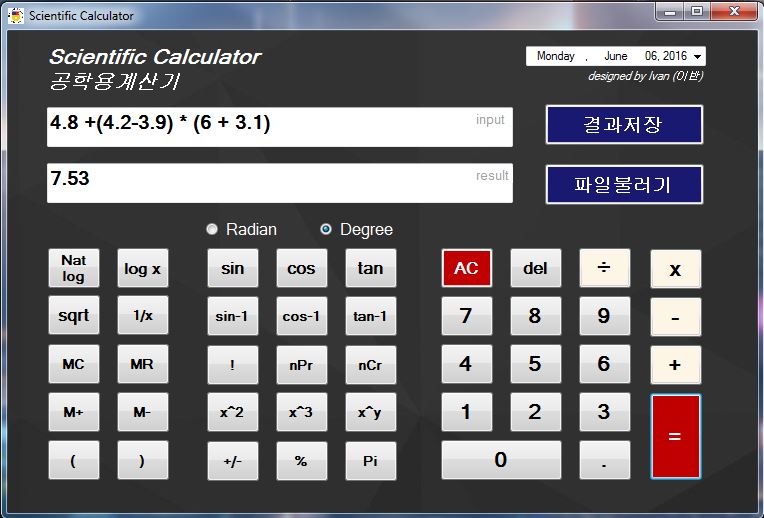


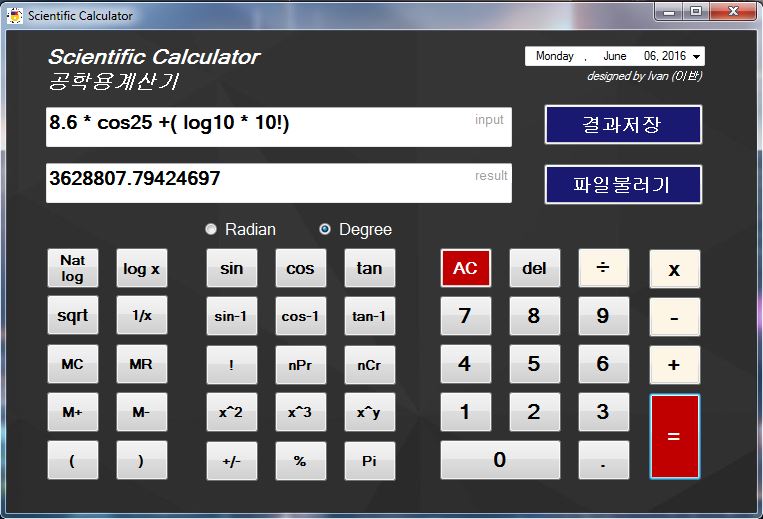


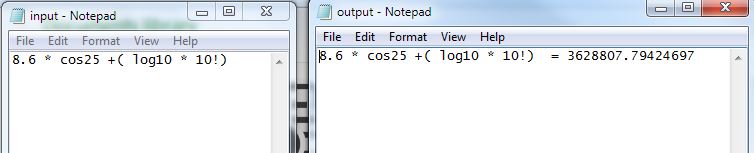
**Scientific Calculator**

**Programming Language : C#**

Scientific calculator project is a GUI application, built using WinForms in C# language. This calculator has most of standard scientific calculator features. Recursive evaluation function is used as the arithmetic / calculation algorithm. This calculator also has input and output features to input arithmetic operation from outer file and save the both of the input and the result to outer file.





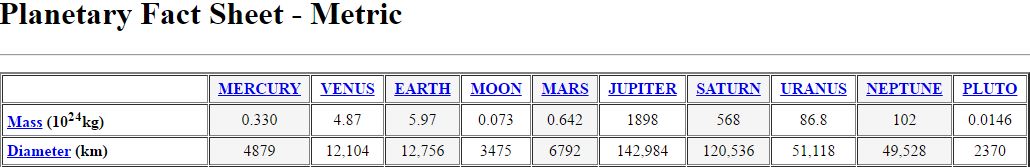


**Computer Graphics : Gravity Simulator**

**Programming Language : C++**

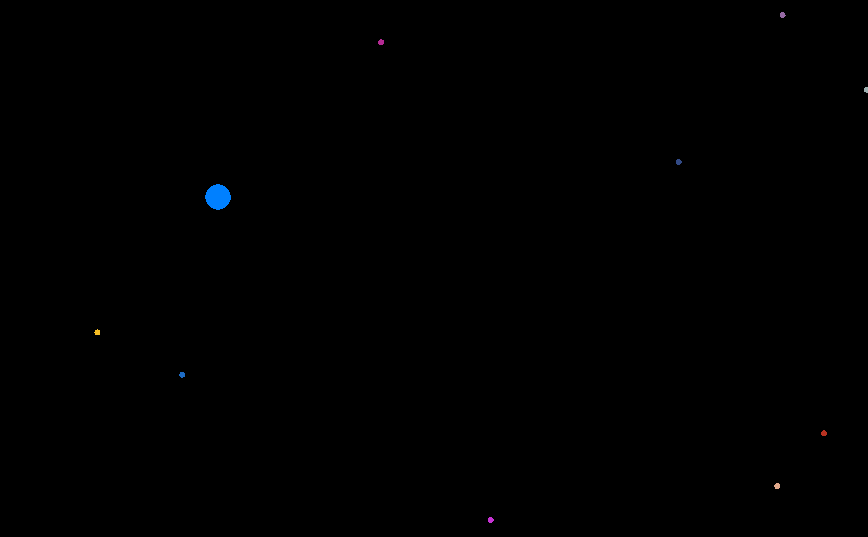
This is a simulation of Newton’s law of universal gravitation using OpenGL in C++. In this project we can see how particles (planets in Milky Way) attract every other particle in the universe using a force that is directly proportional to the product of their masses and inversely proportional to the square of the distance between their centers.

I use the planetary fact sheet provided by NASA to count the mass and diameter of each planet, and apply the Newton’s law of universal gravitation.

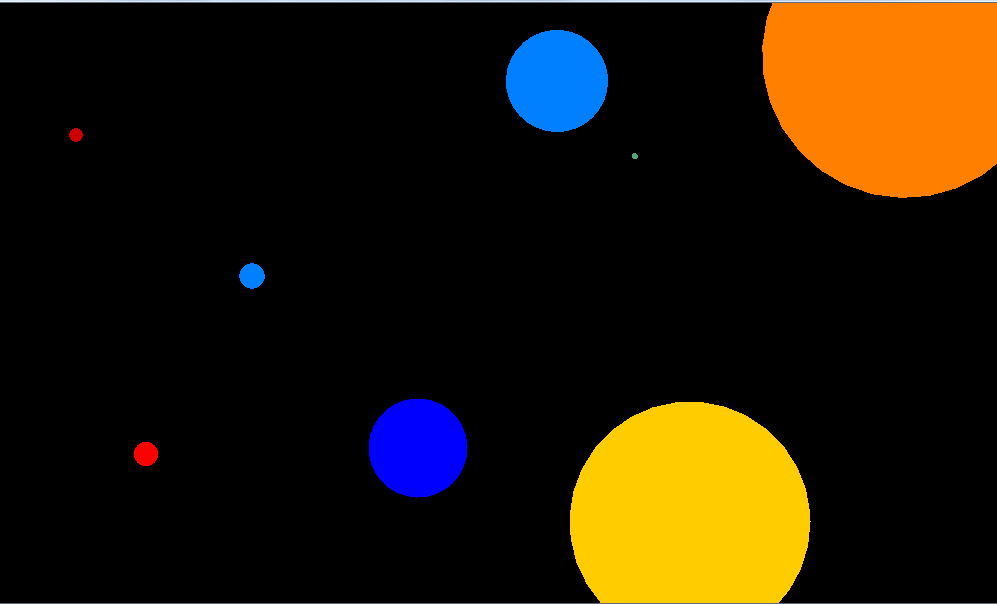


<http://justinduplain.com/nasa-pg/>

In the demonstration video we can see how 1 particle react to other’s particle gravitational force (the stronger the particle’s gravitational force, the faster for other particles to fall down into that particle).



We can add as many small particles that we want (small circle), and big planets in our galaxy (size and color are used to distinguish 1 planet from another). Thing is getting more interesting after we add lots of particles and planets, we can see how the small particles fly across the screen that caused by the gravitational force.



**Microprocessor : Electronic Door Lock**

**Tool : AVR (ATmega 128)**

This is an implementation of microcontroller to create a simple electronic door lock. Like the usual door lock we need to input our own password by pressing the switch button in exact order to lock the system. If we input the wrong password, the LED will flash and the 7-segment will display FALSE. If we input the correct password the LED will flash in different pattern and the 7-segment and LCD will display PASS and the door is opened. To close the system, we just need to press another switch which indicates the door is closed.

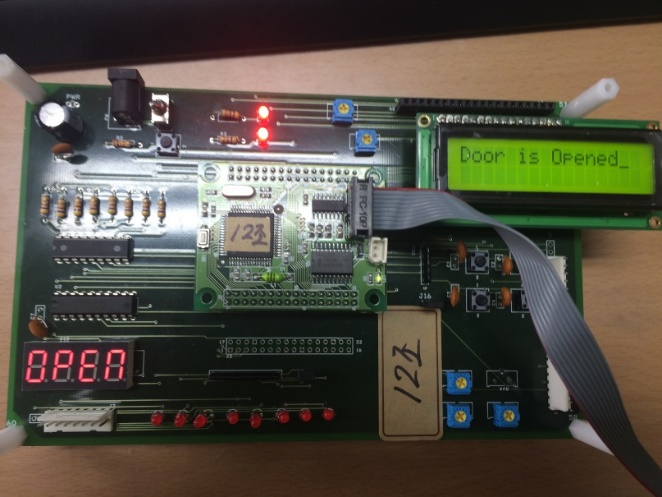


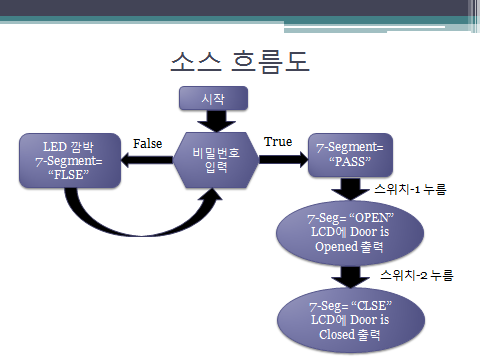
LCD

7-Segment

switch

LED





This project can be extended to create a real door lock if the real required equipment (door, lock) is provided.