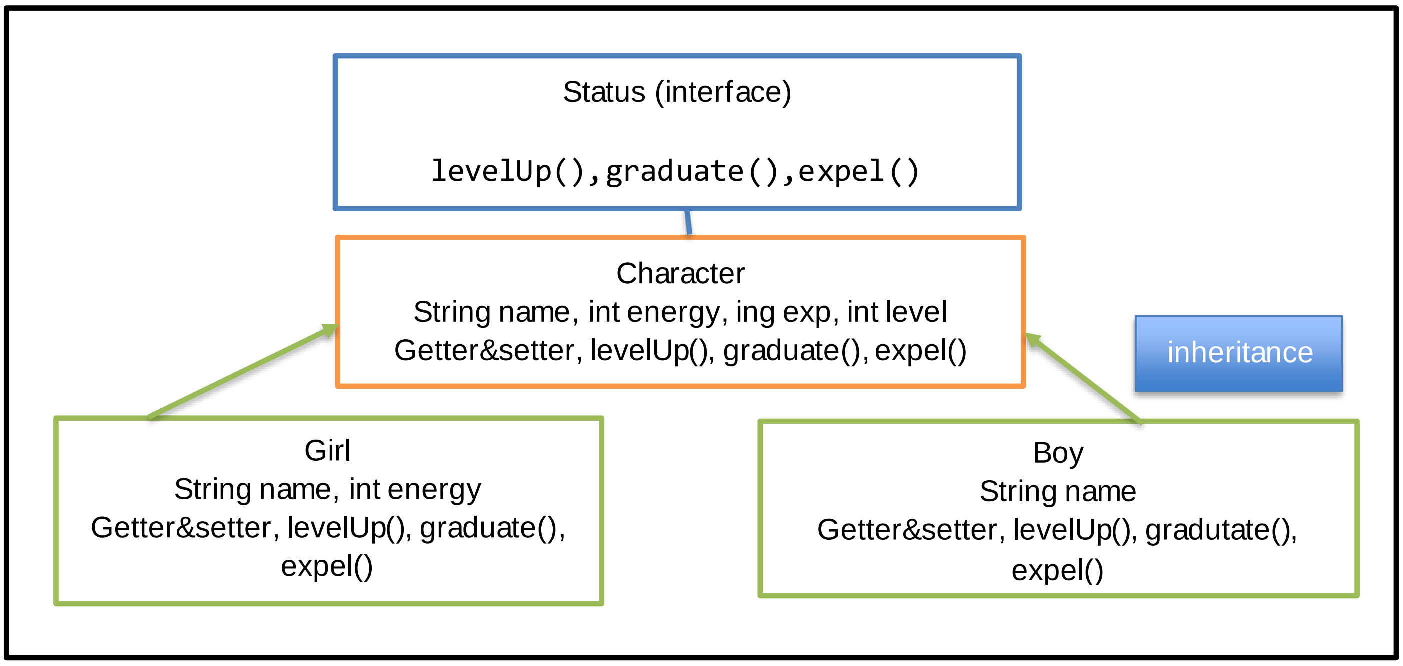
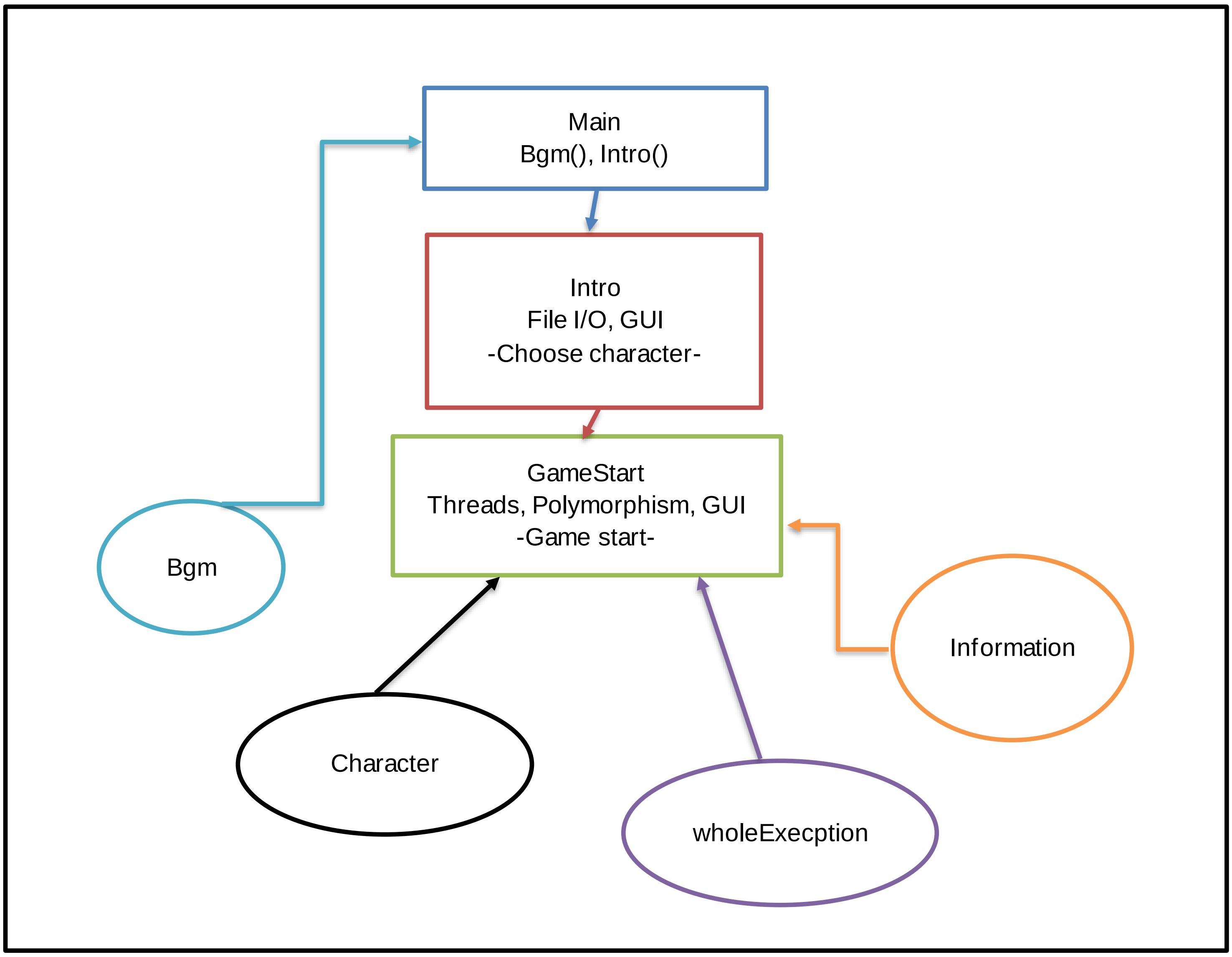
Final Project Report

1. Briefly describe the project purpose:

As a project, I made a game which is Sungkyunkwan University version of Tamagotchi. Tamagotchi is a handheld digital pet that was created in Japan. I thought it would be great to make this game using Sungkyunkwan University mascot. So I made this game for Myeongnyun and Yuljeon, the mascots of Sungkyunkwan University. I think my project could help promote schools in the form of games. In addition, the importance of studying in university can be announced through the contents of the game that if energy and experience values are not properly managed, they can be expelled. In conclusion, the main purpose of this project is to raise and graduate characters well, and additional purposes such as pleasure and school promotion can be obtained.

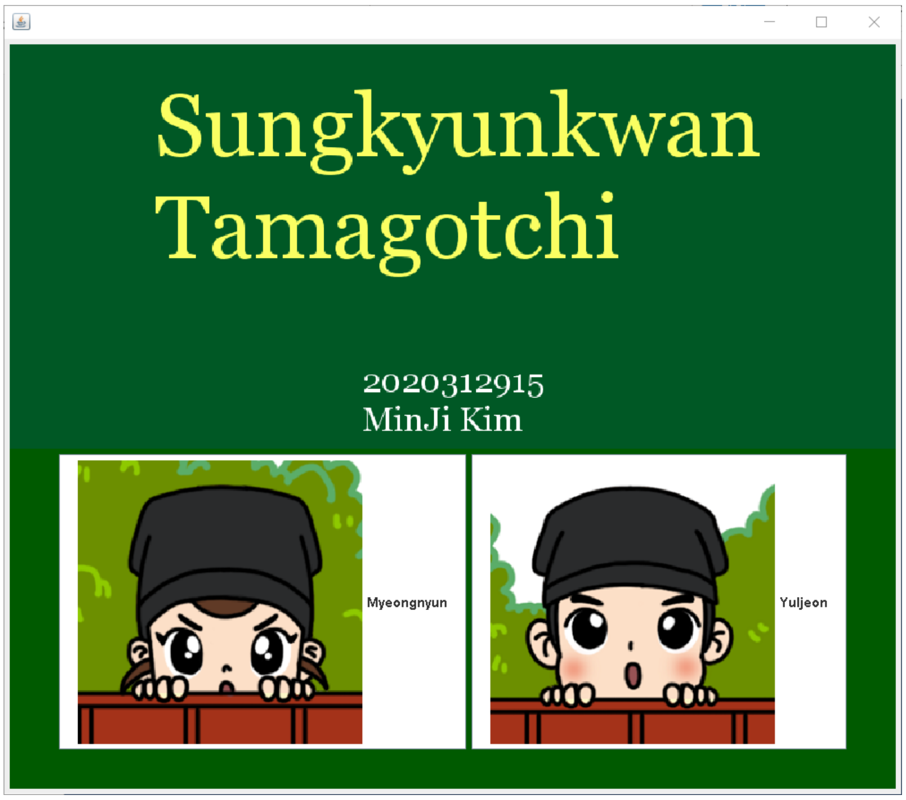
2. Draw the logic flow of the program (with flowchart):


Status (interface)
levelUp(),graduate(),expel()
Character
String name, int energy, ing exp, int level
Getter&setter, levelUp(), graduate(), expel()
inheritance
Girl
String name, int energy
Getter&setter, levelUp(), graduate(), expel()
Boy
String name
Getter&setter, levelUp(), gradutate(), expel()


Main
Bgm(), Intro()
Intro
File I/O, GUI
-Choose character-
GameStart
Threads, Polymorphism, GUI
-Game start-
Bgm
Information
Character
wholeExecption

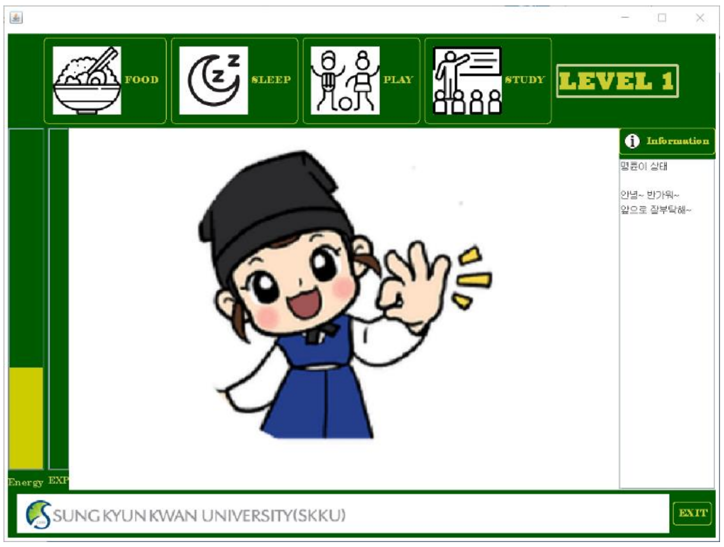
3. Provide screenshots for each screen with brief description:

1) start screen



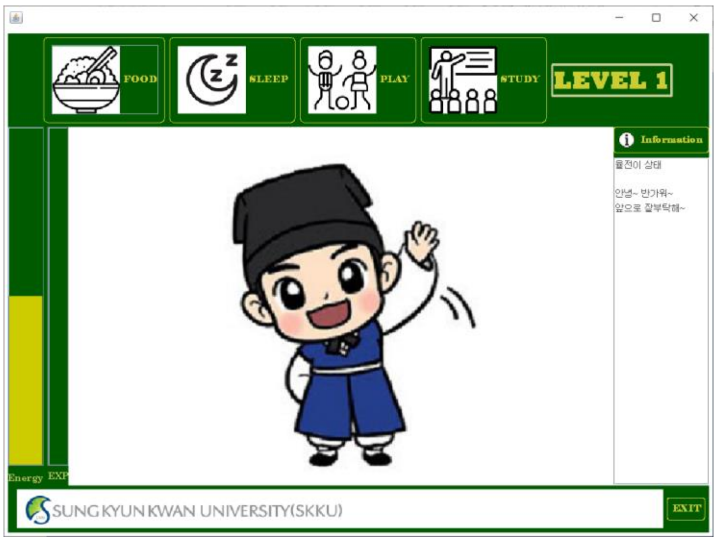
This screen is the start screen and there are two buttons. Users can choose between Myeongryun and Yuljeon. Also, music plays as soon as the game starts.

2) main screen



명륜이 상태

안녕~ 반가워~
앞으로 잘부탁해~

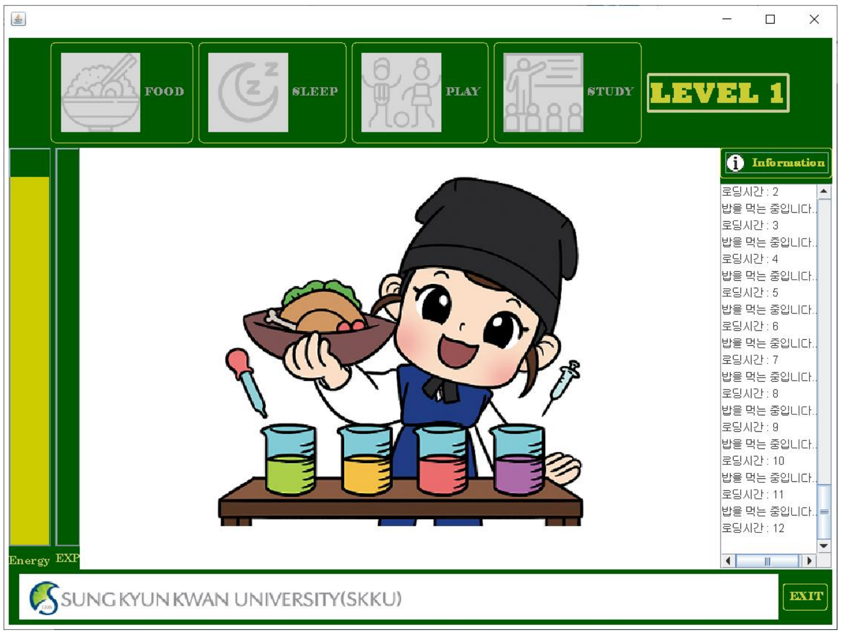


율전이 상태

안녕~ 반가워~
앞으로 잘부탁해~

When the user selects a character, the main screen appears and a greeting is output in the status window. In addition, the initial energy values of the characters appear in the energy bar. It can be seen that the initial energy values of the two characters are different.

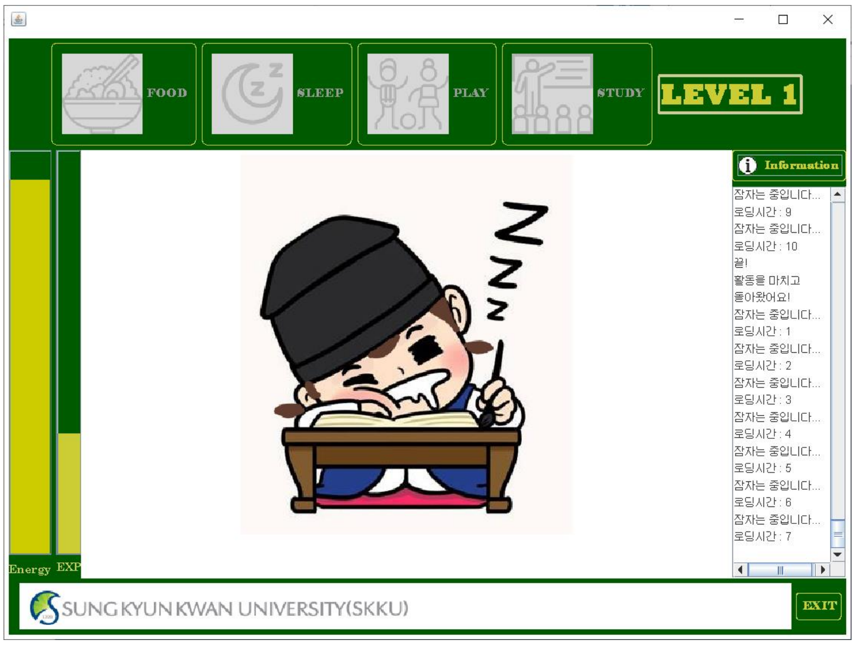
3) Food screen



```
로딩시간 : 2
밥을 먹는 중입니다..
로딩시간 : 3
밥을 먹는 중입니다..
로딩시간 : 4
밥을 먹는 중입니다..
로딩시간 : 5
밥을 먹는 중입니다..
로딩시간 : 6
밥을 먹는 중입니다..
로딩시간 : 7
밥을 먹는 중입니다..
로딩시간 : 8
밥을 먹는 중입니다..
로딩시간 : 9
밥을 먹는 중입니다..
로딩시간 : 10
밥을 먹는 중입니다..
로딩시간 : 11
밥을 먹는 중입니다..
로딩시간 : 12
```

If user presses FOOD button, energy increases by 25. In addition, a statement about the loading time is output in the status window, and the energy bar is raised. When the character is eating food, the activity buttons become unusable.

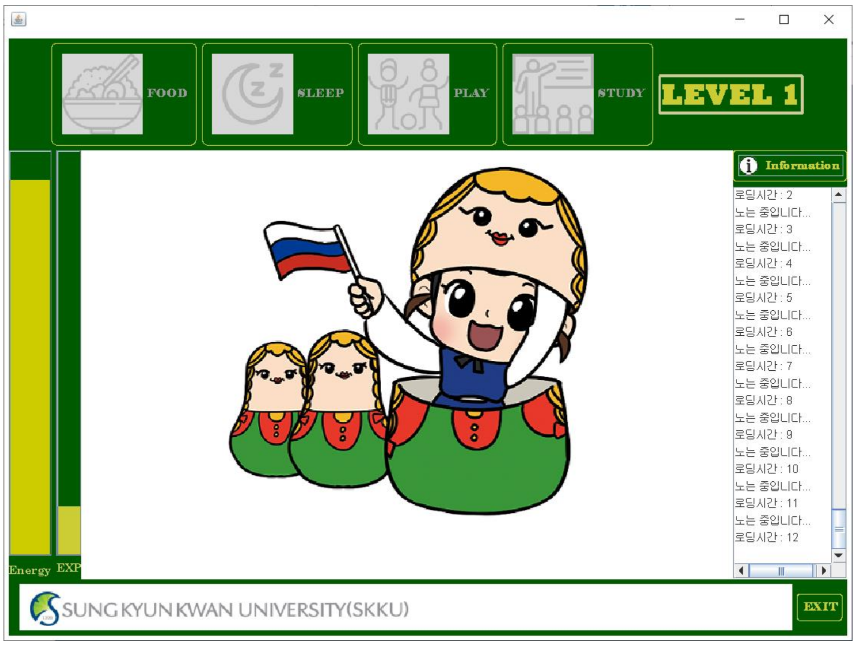
4) Sleep screen



```
잠자는 중입니다...
로딩시간 : 9
잠자는 중입니다...
로딩시간 : 10
끝!
활동을 마치고
돌아왔어요!
잠자는 중입니다...
로딩시간 : 1
잠자는 중입니다...
로딩시간 : 2
잠자는 중입니다...
로딩시간 : 3
잠자는 중입니다...
로딩시간 : 4
잠자는 중입니다...
로딩시간 : 5
잠자는 중입니다...
로딩시간 : 6
잠자는 중입니다...
로딩시간 : 7
```

If user presses SLEEP button, energy increases by 10. In addition, a statement about the loading time is output in the status window, and the energy bar is raised. When the character is sleeping, the activity buttons become unusable.

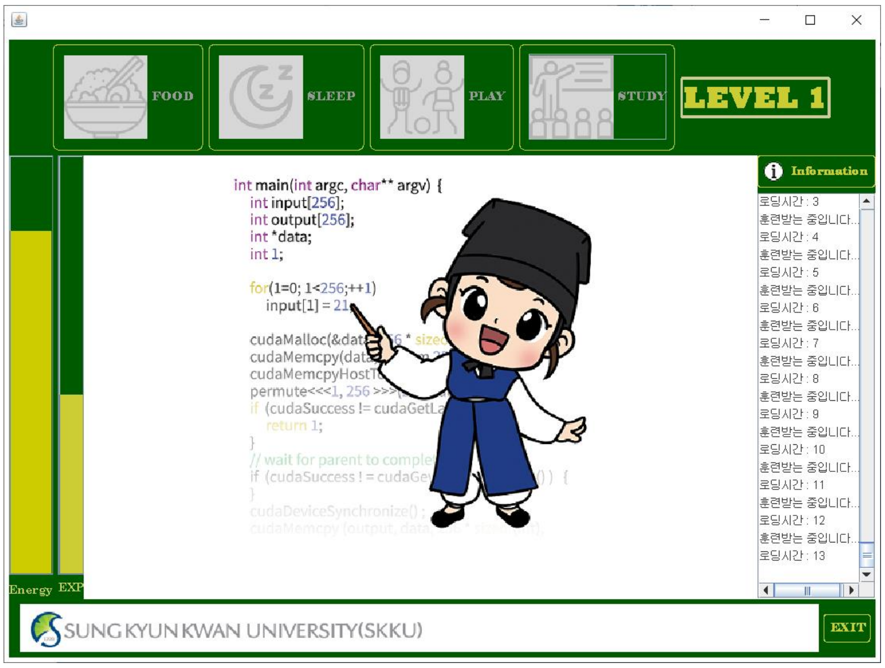
5) Play screen



```
로딩시간 : 2
노는 중입니다...
로딩시간 : 3
노는 중입니다...
로딩시간 : 4
노는 중입니다...
로딩시간 : 5
노는 중입니다...
로딩시간 : 6
노는 중입니다...
로딩시간 : 7
노는 중입니다...
로딩시간 : 8
노는 중입니다...
로딩시간 : 9
노는 중입니다...
로딩시간 : 10
노는 중입니다...
로딩시간 : 11
노는 중입니다...
로딩시간 : 12
```

If user presses PLAY button, energy decreases by 30 and exp(experience) increases by 30. In addition, a statement about the loading time is output in the status window, and the energy bar reduces and exp bar is raised. When the character is playing, the activity buttons become unusable.

6) Study screen

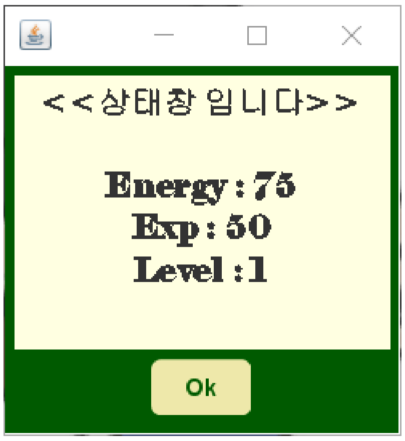


```
int main(int argc, char** argv) {
   int input[256];
   int output[256];
   int *data;
   int 1;

   for(1=0; 1<256;++1)
      input[1] = 21

   cudaMalloc(&dat  6 * size
   cudaMemcpy(data
   cudaMemcpyHostT
   permute<<<1, 256 >>>
   if (cudaSuccess != cudaGetLa
      return 1;
   }
   // wait for parent to complet
   if (cudaSuccess ! = cudaGe              ()) {
   }
   cudaDeviceSynchronize() ;
```

```
로딩시간 : 3
훈련받는 중입니다...
로딩시간 : 4
훈련받는 중입니다...
로딩시간 : 5
훈련받는 중입니다...
로딩시간 : 6
훈련받는 중입니다...
로딩시간 : 7
훈련받는 중입니다...
로딩시간 : 8
훈련받는 중입니다...
로딩시간 : 9
훈련받는 중입니다...
로딩시간 : 10
훈련받는 중입니다...
로딩시간 : 11
훈련받는 중입니다...
로딩시간 : 12
훈련받는 중입니다...
로딩시간 : 13
```

If user presses STUDY button, energy decreases by 20 and exp(experience) increases by 20. In addition, a statement about the loading time is output in the status window, and the energy bar reduces and exp bar is raised. When the character is studying, the activity buttons become unusable.

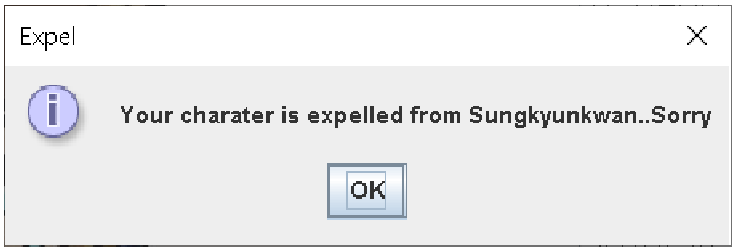
7) Information screen


<<상태창 입니다>>

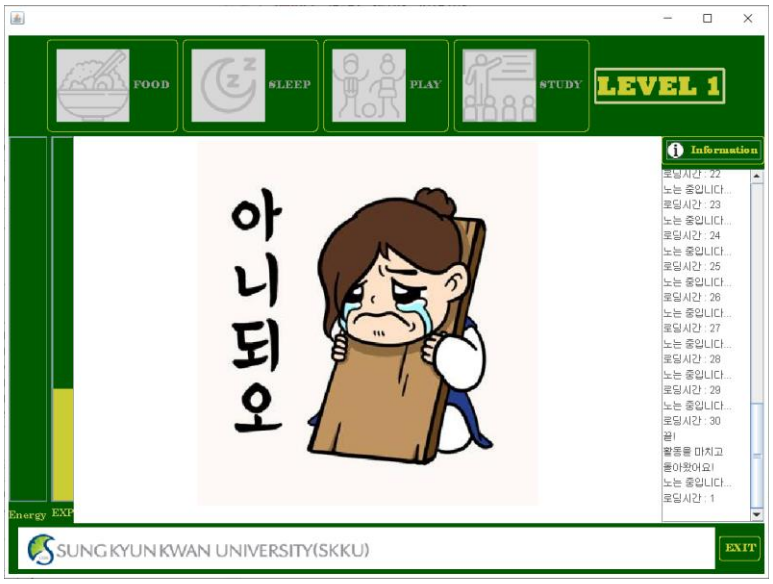
Energy : 75
Exp : 50
Level : 1

This is the status window and it comes out when user presses the information button at the right side. It shows the character's current energy, exp, and level. It closes when you press the OK button.

8) Expel screen


Expel

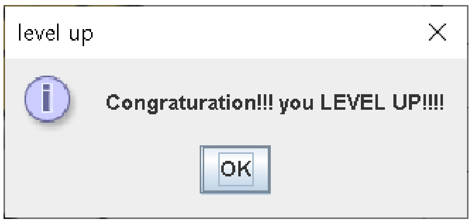
Your charater is expelled from Sungkyunkwan..Sorry



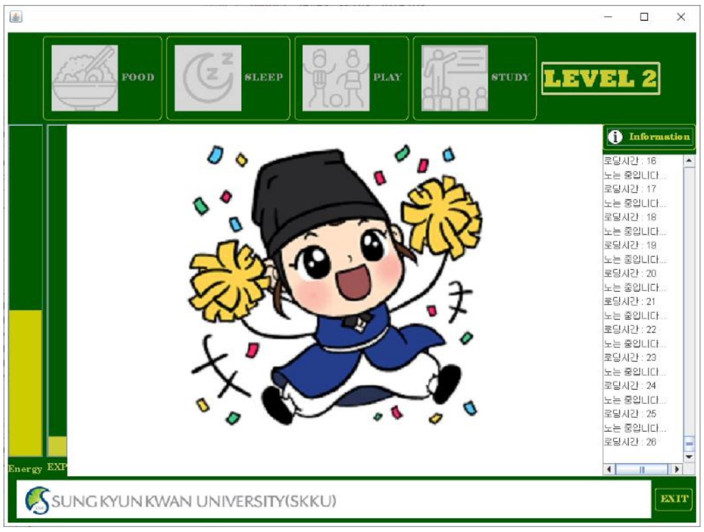
```
로딩시간 : 22
노는 중입니다...
로딩시간 : 23
노는 중입니다...
로딩시간 : 24
노는 중입니다...
로딩시간 : 25
노는 중입니다...
로딩시간 : 26
노는 중입니다...
로딩시간 : 27
노는 중입니다...
로딩시간 : 28
노는 중입니다...
로딩시간 : 29
노는 중입니다...
로딩시간 : 30
끝!
활동을 마치고
돌아왔어요!
노는 중입니다...
로딩시간 : 1
```

If the energy drops below zero, a message pops up indicating that the student has been expelled and the game ends.

9) Level up screen


level up

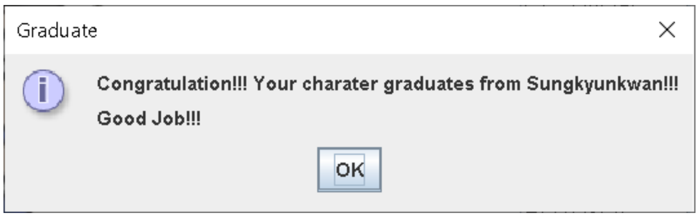
Congraturation!!! you LEVEL UP!!!!



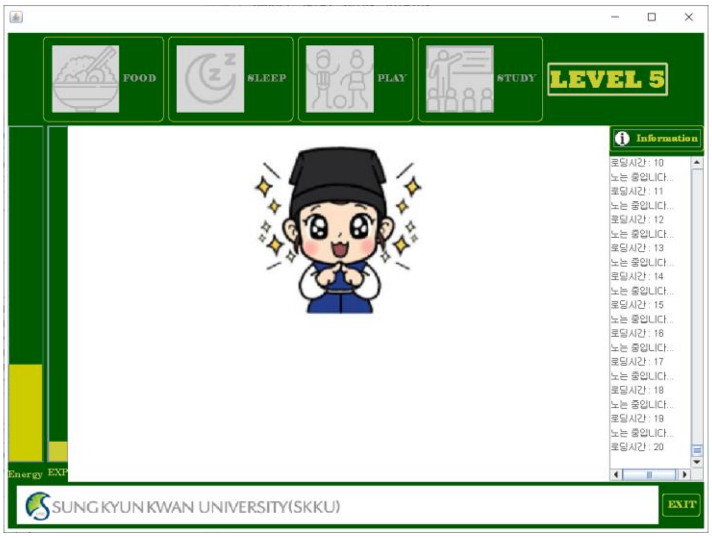
```
로딩시간 : 16
노는 중입니다...
로딩시간 : 17
노는 중입니다...
로딩시간 : 18
노는 중입니다...
로딩시간 : 19
노는 중입니다...
로딩시간 : 20
노는 중입니다...
로딩시간 : 21
노는 중입니다...
로딩시간 : 22
노는 중입니다...
로딩시간 : 23
노는 중입니다...
로딩시간 : 24
노는 중입니다...
로딩시간 : 25
노는 중입니다...
로딩시간 : 26
```

If exp exceeds 80, the level up message window is displayed and the increased level is reflected on the label and displayed.

10) Graduate screen


Graduate

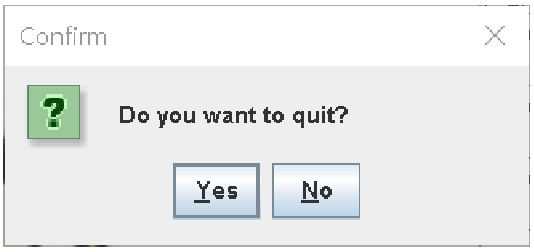
Congratulation!!! Your charater graduates from Sungkyunkwan!!!
Good Job!!!



```
로딩시간 : 10
노는 중입니다...
로딩시간 : 11
노는 중입니다...
로딩시간 : 12
노는 중입니다...
로딩시간 : 13
노는 중입니다...
로딩시간 : 14
노는 중입니다...
로딩시간 : 15
노는 중입니다...
로딩시간 : 16
노는 중입니다...
로딩시간 : 17
노는 중입니다...
로딩시간 : 18
노는 중입니다...
로딩시간 : 19
노는 중입니다...
로딩시간 : 20
```

If the level reaches 5, a message window appears to congratulate character on graduation and the game ends.

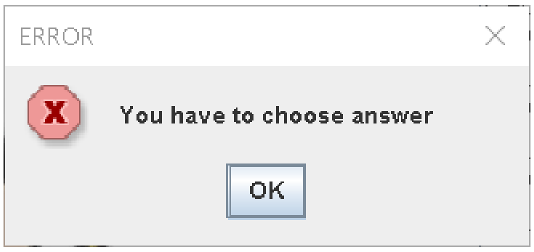
11) Exit screen


Confirm

Do you want to quit?

If user presses the exit button on the bottom right, users gets a confirmation message. When the user presses yes, the game ends, and when the user presses no, the message window disappears and user can continue the game.

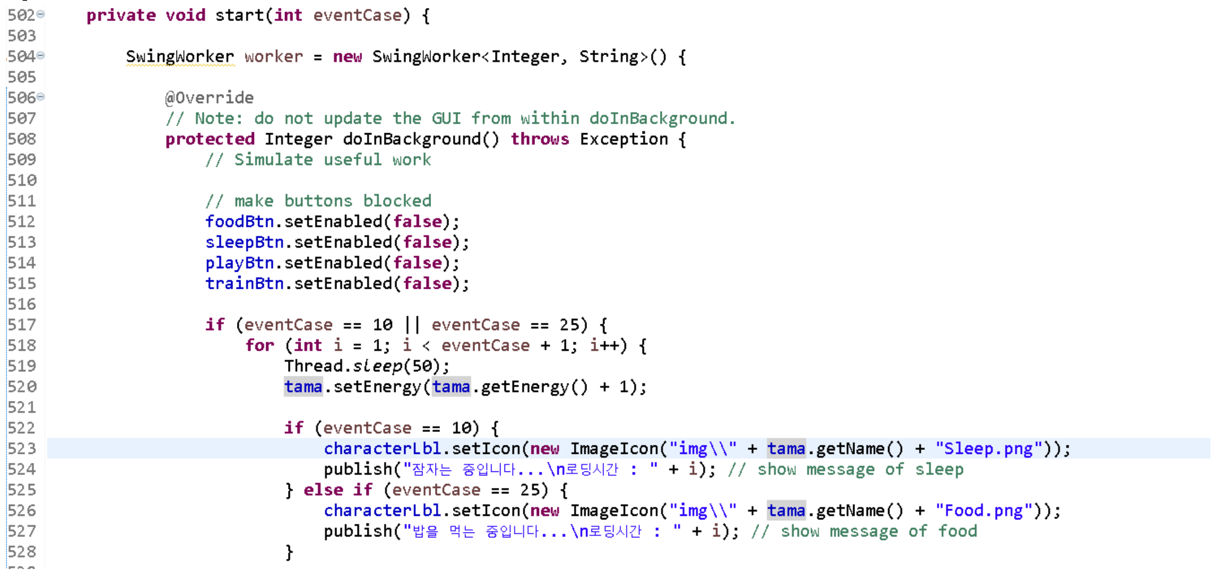
12) Error screen


ERROR

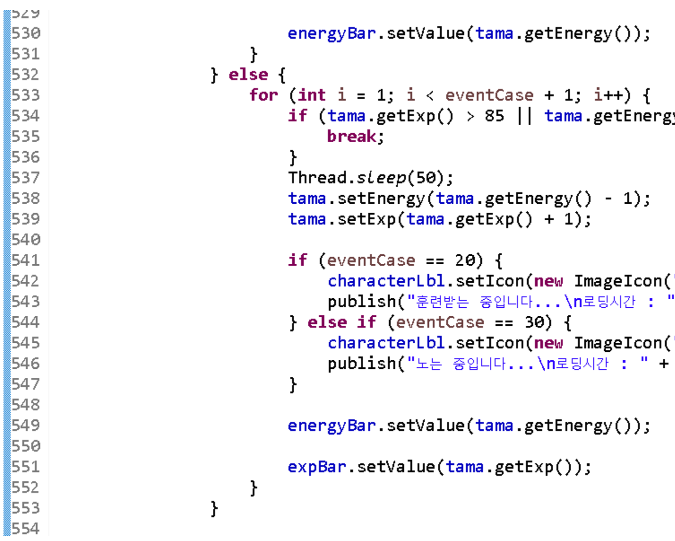
You have to choose answer

If the user presses exit without selecting yes or no, an error message appears.

4. Explain the code of the main functionalities

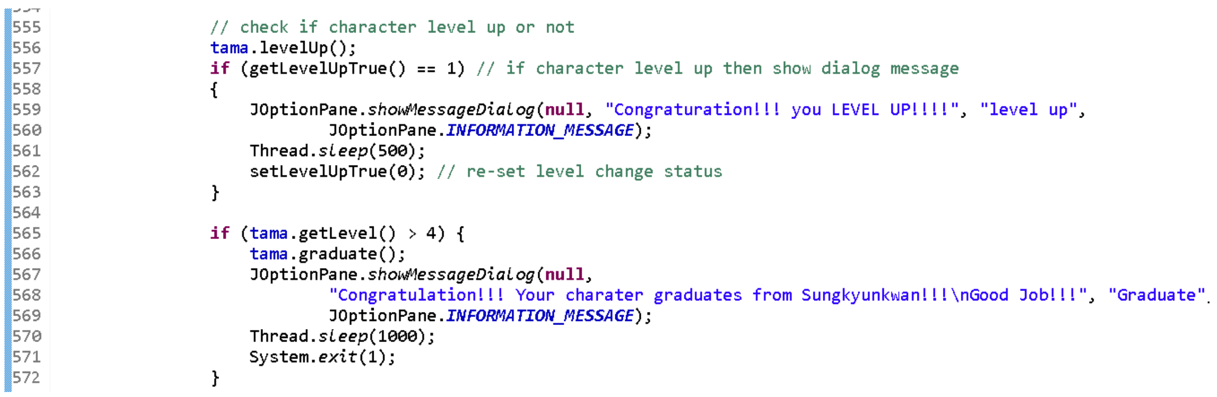


```java
private void start(int eventCase) {

    SwingWorker worker = new SwingWorker<Integer, String>() {

        @Override
        // Note: do not update the GUI from within doInBackground.
        protected Integer doInBackground() throws Exception {
            // Simulate useful work

            // make buttons blocked
            foodBtn.setEnabled(false);
            sleepBtn.setEnabled(false);
            playBtn.setEnabled(false);
            trainBtn.setEnabled(false);

            if (eventCase == 10 || eventCase == 25) {
                for (int i = 1; i < eventCase + 1; i++) {
                    Thread.sleep(50);
                    tama.setEnergy(tama.getEnergy() + 1);

                    if (eventCase == 10) {
                        characterLbl.setIcon(new ImageIcon("img\\" + tama.getName() + "Sleep.png"));
                        publish("잠자는 중입니다...\n로딩시간 : " + i); // show message of sleep
                    } else if (eventCase == 25) {
                        characterLbl.setIcon(new ImageIcon("img\\" + tama.getName() + "Food.png"));
                        publish("밥을 먹는 중입니다...\n로딩시간 : " + i); // show message of food
                    }
```

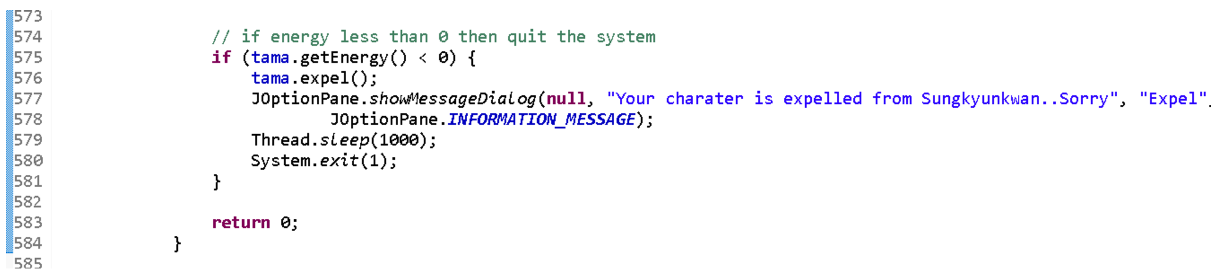


```java

                    energyBar.setValue(tama.getEnergy());
                }
            } else {
                for (int i = 1; i < eventCase + 1; i++) {
                    if (tama.getExp() > 85 || tama.getEnergy() < 0) {
                        break;
                    }
                    Thread.sleep(50);
                    tama.setEnergy(tama.getEnergy() - 1);
                    tama.setExp(tama.getExp() + 1);

                    if (eventCase == 20) {
                        characterLbl.setIcon(new ImageIcon("img\\" + tama.getName() + "Train.png"));
                        publish("훈련받는 중입니다...\n로딩시간 : " + i); // show message of train
                    } else if (eventCase == 30) {
                        characterLbl.setIcon(new ImageIcon("img\\" + tama.getName() + "Play.png"));
                        publish("노는 중입니다...\n로딩시간 : " + i); // show message of play
                    }

                    energyBar.setValue(tama.getEnergy());

                    expBar.setValue(tama.getExp());
                }
            }

```

The start function is a function that increases or decreases energy and exp. It was divided by case and publish String each case so that the loading string could appear on the screen. It also changes the value of progress bar in the iteration statement.

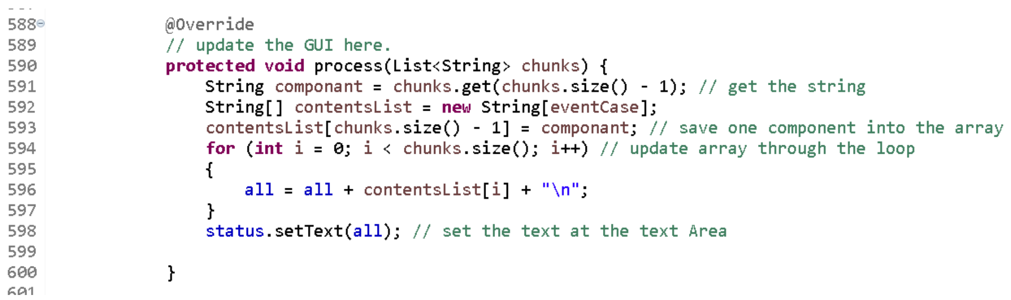


```java
            // check if character level up or not
            tama.levelUp();
            if (getLevelUpTrue() == 1) // if character level up then show dialog message
            {
                JOptionPane.showMessageDialog(null, "Congraturation!!! you LEVEL UP!!!!", "level up",
                        JOptionPane.INFORMATION_MESSAGE);
                Thread.sleep(500);
                setLevelUpTrue(0); // re-set level change status
            }

            if (tama.getLevel() > 4) {
                tama.graduate();
                JOptionPane.showMessageDialog(null,
                        "Congratulation!!! Your charater graduates from Sungkyunkwan!!!\nGood Job!!!", "Graduate"
                        JOptionPane.INFORMATION_MESSAGE);
                Thread.sleep(1000);
                System.exit(1);
            }
```



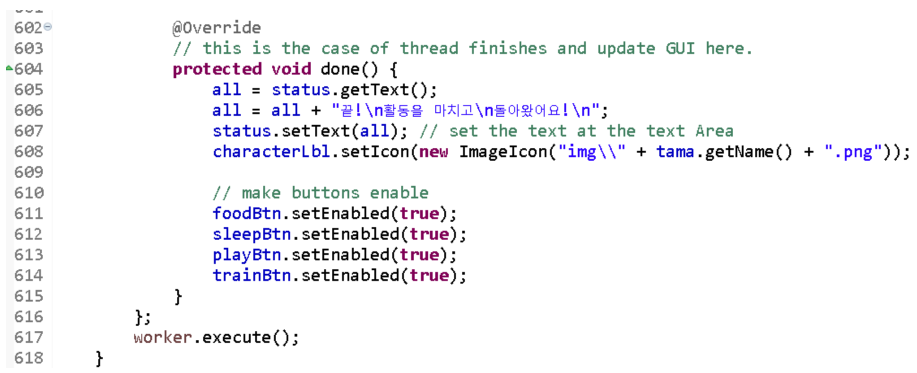
```java
            // if energy less than 0 then quit the system
            if (tama.getEnergy() < 0) {
                tama.expel();
                JOptionPane.showMessageDialog(null, "Your charater is expelled from Sungkyunkwan..Sorry", "Expel"
                        JOptionPane.INFORMATION_MESSAGE);
                Thread.sleep(1000);
                System.exit(1);
            }

            return 0;
        }
```

Check the level and energy status after the repeat statement. After checking the status, a message window is displayed according to its status.



```java
        @Override
        // update the GUI here.
        protected void process(List<String> chunks) {
            String componant = chunks.get(chunks.size() - 1); // get the string
            String[] contentsList = new String[eventCase];
            contentsList[chunks.size() - 1] = componant; // save one component into the array
            for (int i = 0; i < chunks.size(); i++) // update array through the loop
            {
                all = all + contentsList[i] + "\n";
            }
            status.setText(all); // set the text at the text Area

        }
```

In the process part, the text thrown through publish is displayed on the screen.

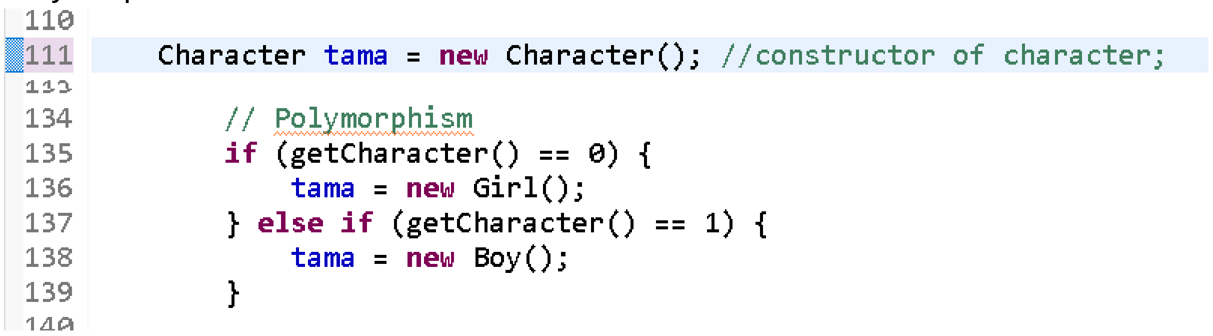


```java
            @Override
            // this is the case of thread finishes and update GUI here.
            protected void done() {
                all = status.getText();
                all = all + "끝!\n활동을 마치고\n돌아왔어요!\n";
                status.setText(all); // set the text at the text Area
                characterLbl.setIcon(new ImageIcon("img\\" + tama.getName() + ".png"));

                // make buttons enable
                foodBtn.setEnabled(true);
                sleepBtn.setEnabled(true);
                playBtn.setEnabled(true);
                trainBtn.setEnabled(true);
            }
        };
        worker.execute();
    }
```

When the entire process is complete, the end message is displayed on the screen and the button is made available again.

5. Explain what is included in your project and why it is used (Polymorphism, Inheritance, File I/O, etc)

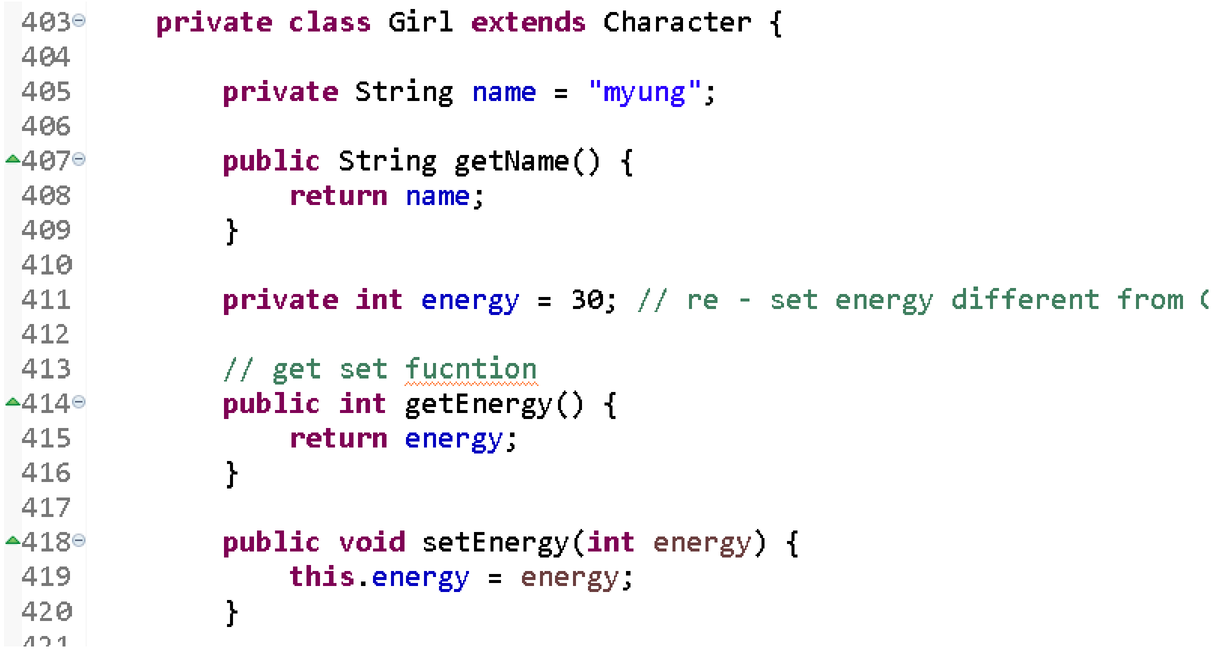
1) Polymorphism



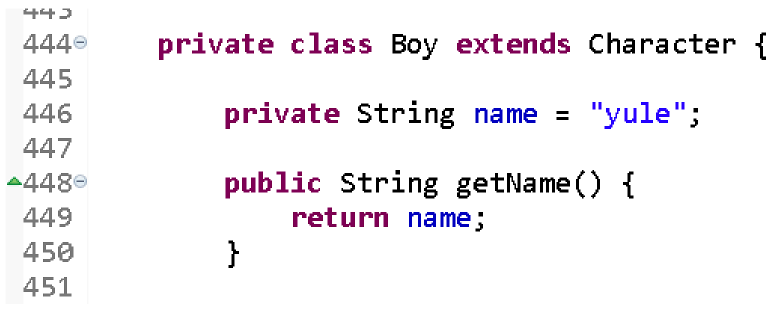
```
    Character tama = new Character(); //constructor of character;

        // Polymorphism
        if (getCharacter() == 0) {
            tama = new Girl();
        } else if (getCharacter() == 1) {
            tama = new Boy();
        }
```

Since there are two characters in my game, polymorphism was required to operate the code in both cases with one variable name. In addition, the initial energy values of Myeongnyun and Yuljeon are different, so it was used to reflect this.

2) Inheritance



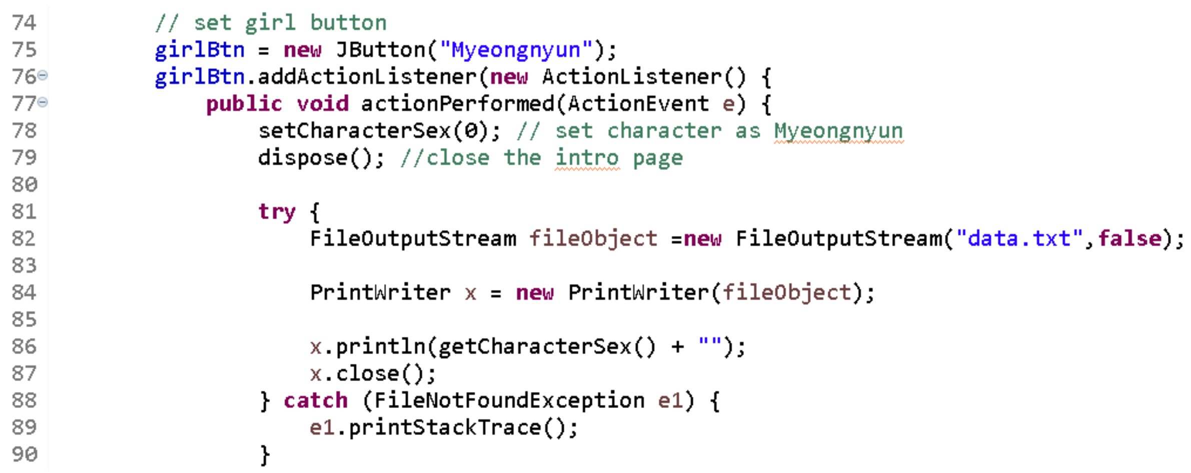
```
    private class Girl extends Character {

        private String name = "myung";

        public String getName() {
            return name;
        }

        private int energy = 30; // re - set energy different from (

        // get set fucntion
        public int getEnergy() {
            return energy;
        }

        public void setEnergy(int energy) {
            this.energy = energy;
        }
```



```
    private class Boy extends Character {

        private String name = "yule";

        public String getName() {
            return name;
        }

```

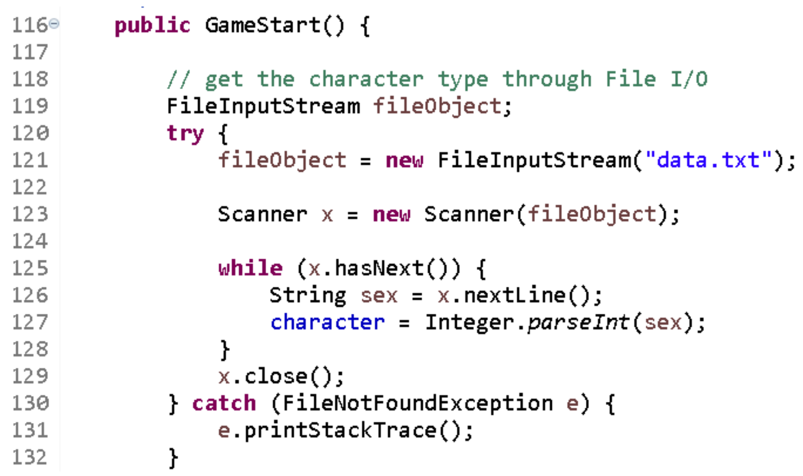
These two classes share the exp and level of the character class. In addition, the function of the character class is overridden and used. Therefore, an implementation was used to save the number of variables which are used in this project.

3) File I/O



```java
        // set girl button
        girlBtn = new JButton("Myeongnyun");
        girlBtn.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                setCharacterSex(0); // set character as Myeongnyun
                dispose(); //close the intro page

                try {
                    FileOutputStream fileObject =new FileOutputStream("data.txt",false);

                    PrintWriter x = new PrintWriter(fileObject);

                    x.println(getCharacterSex() + "");
                    x.close();
                } catch (FileNotFoundException e1) {
                    e1.printStackTrace();
                }
```

When choosing a character on the start screen (Intro), if the user chooses Myeongryun then save 0 in the data file. If the user chooses Yuljeon, 1 is stored in the data file.



```java
    public GameStart() {

        // get the character type through File I/O
        FileInputStream fileObject;
        try {
            fileObject = new FileInputStream("data.txt");

            Scanner x = new Scanner(fileObject);

            while (x.hasNext()) {
                String sex = x.nextLine();
                character = Integer.parseInt(sex);
            }
            x.close();
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
```

At the start of the game, the number is read from the stored data file to determine whether it is Myeongryun or Yuljeon.

You can add more sections based on your project.