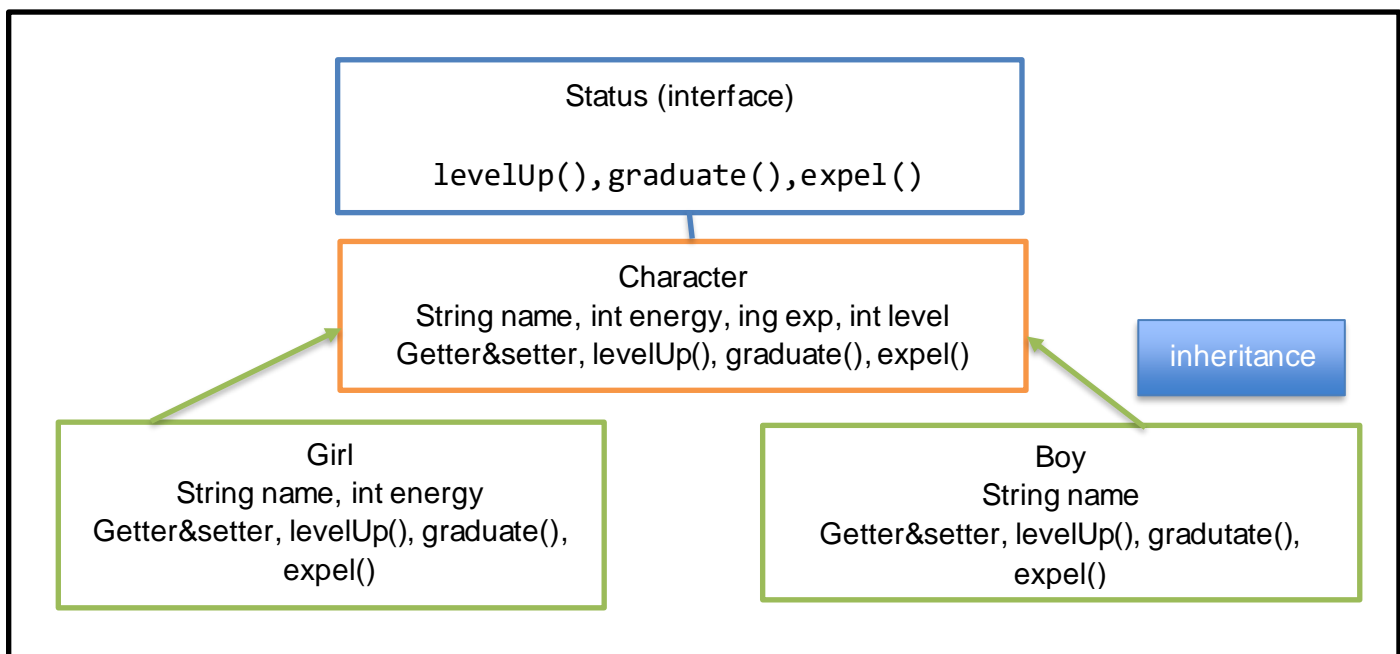


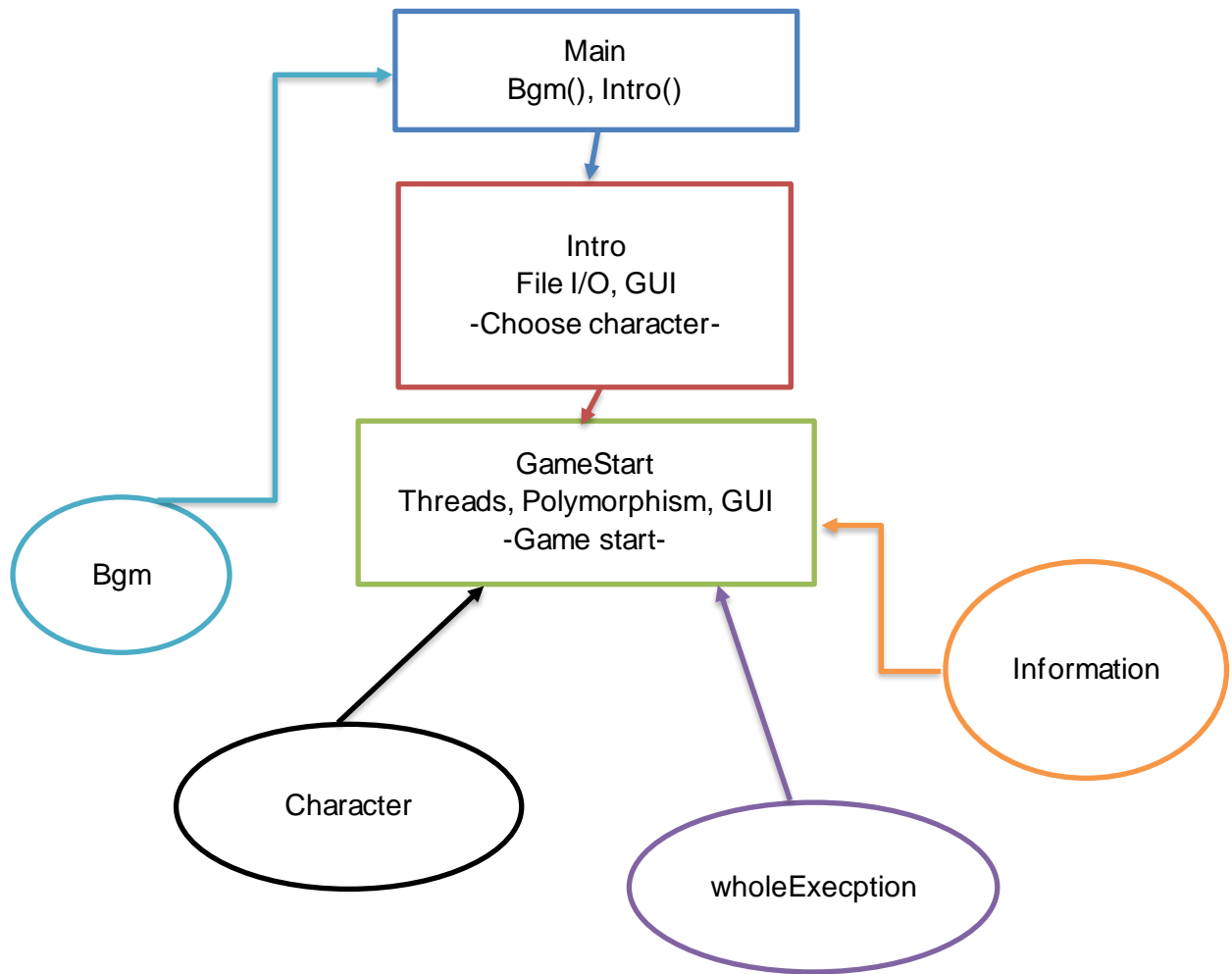
Final Project Report

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):





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



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



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



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



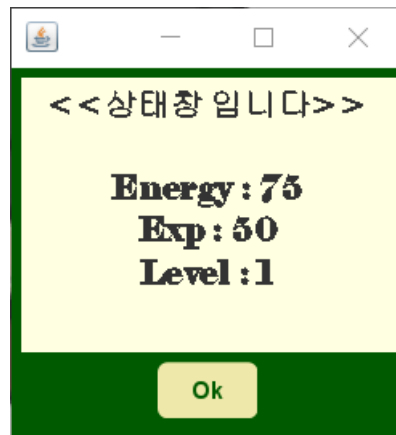
If user presses PLAY button, energy decreases by 30 and exp(experience) increases by 30. In addition, a statement about the loading time is output in the status window, and the energy bar reduces and exp bar is raised. When the character is playing, the activity buttons become unusable.

6) Study screen



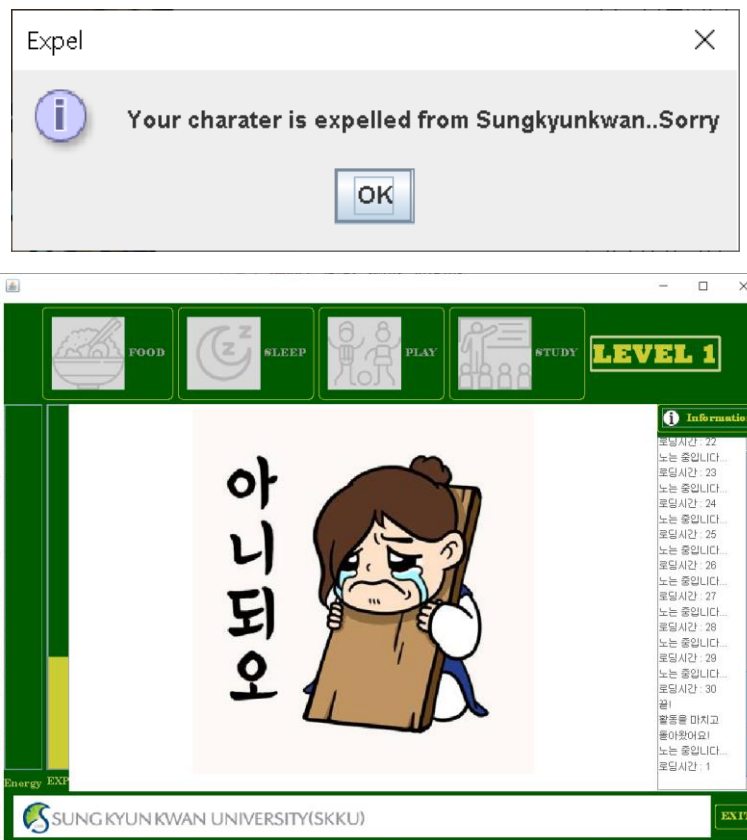
If user presses STUDY button, energy decreases by 20 and exp(experience) increases by 20. In addition, a statement about the loading time is output in the status window, and the energy bar reduces and exp bar is raised. When the character is studying, the activity buttons become unusable.

7) Information screen



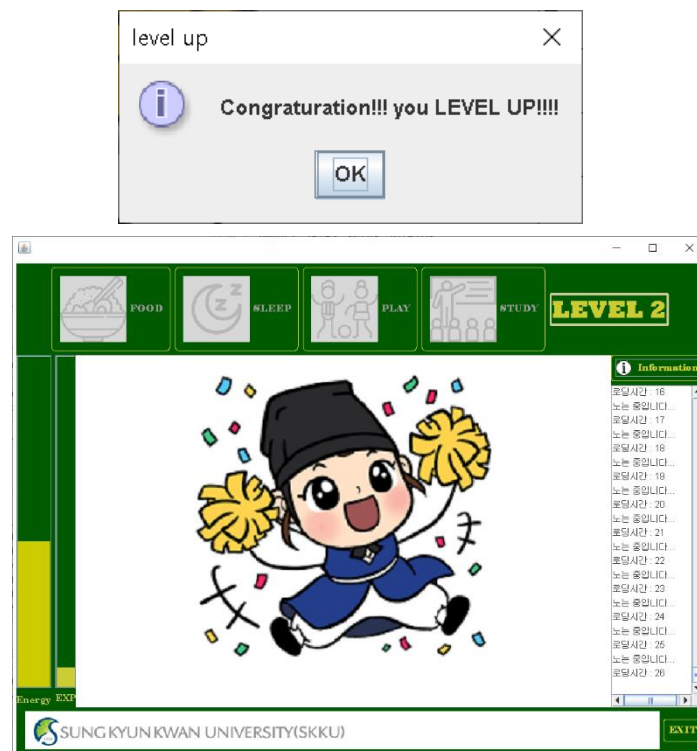
This is the status window and it comes out when user presses the information button at the right side. It shows the character's current energy, exp, and level. It closes when you press the OK button.

8) Expel screen



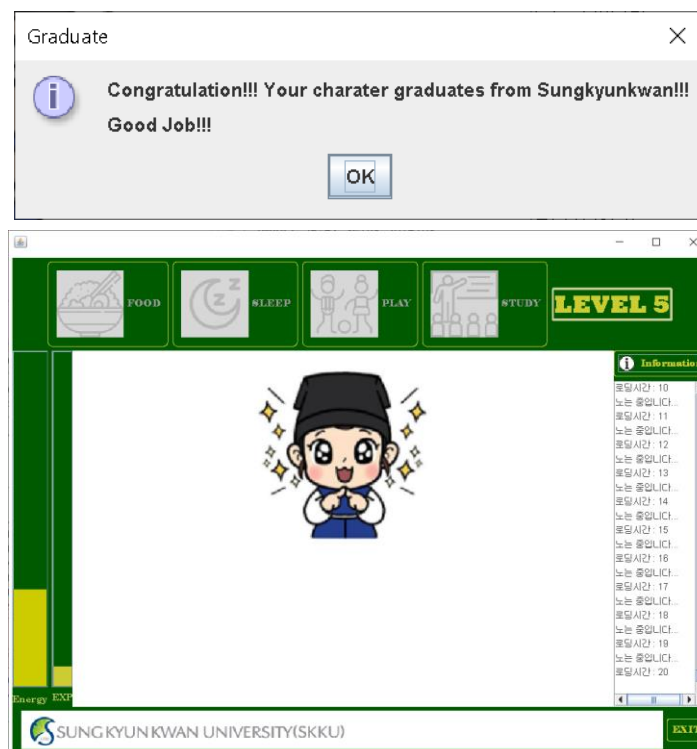
If the energy drops below zero, a message pops up indicating that the student has been expelled and the game ends.

9) Level up screen



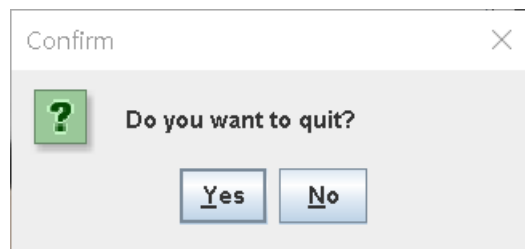
If exp exceeds 80, the level up message window is displayed and the increased level is reflected on the label and displayed.

10) Graduate screen



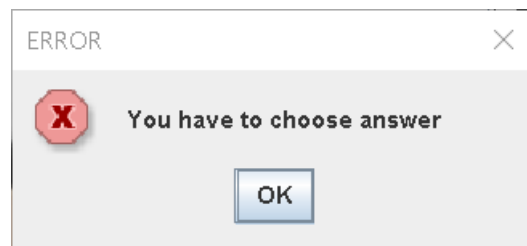
If the level reaches 5, a message window appears to congratulate character on graduation and the game ends.

11) Exit screen



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



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

4. Explain the code of the main functionalities

```
502 private void start(int eventCase) {
503
504     SwingWorker worker = new SwingWorker<Integer, String>() {
505
506         @Override
507         // Note: do not update the GUI from within doInBackground.
508         protected Integer doInBackground() throws Exception {
509             // Simulate useful work
510
511             // make buttons blocked
512             foodBtn.setEnabled(false);
513             sleepBtn.setEnabled(false);
514             playBtn.setEnabled(false);
515             trainBtn.setEnabled(false);
516
517             if (eventCase == 10 || eventCase == 25) {
518                 for (int i = 1; i < eventCase + 1; i++) {
519                     Thread.sleep(50);
520                     tama.setEnergy(tama.getEnergy() + 1);
521
522                     if (eventCase == 10) {
523                         characterLbl.setIcon(new ImageIcon("img\\" + tama.getName() + "Sleep.png"));
524                         publish("잠자는 중입니다...\n로딩시간 : " + i); // show message of sleep
525                     } else if (eventCase == 25) {
526                         characterLbl.setIcon(new ImageIcon("img\\" + tama.getName() + "Food.png"));
527                         publish("밥을 먹는 중입니다...\n로딩시간 : " + i); // show message of food
528                     }
529
530                     energyBar.setValue(tama.getEnergy());
531                 }
532             } else {
533                 for (int i = 1; i < eventCase + 1; i++) {
534                     if (tama.getExp() > 85 || tama.getEnergy() < 0) {
535                         break;
536                     }
537                     Thread.sleep(50);
538                     tama.setEnergy(tama.getEnergy() - 1);
539                     tama.setExp(tama.getExp() + 1);
540
541                     if (eventCase == 20) {
542                         characterLbl.setIcon(new ImageIcon("img\\" + tama.getName() + "Train.png"));
543                         publish("훈련받는 중입니다...\n로딩시간 : " + i); // show message of train
544                     } else if (eventCase == 30) {
545                         characterLbl.setIcon(new ImageIcon("img\\" + tama.getName() + "Play.png"));
546                         publish("노는 중입니다...\n로딩시간 : " + i); // show message of play
547                     }
548
549                     energyBar.setValue(tama.getEnergy());
550
551                     expBar.setValue(tama.getExp());
552                 }
553             }
554         }
555     }
556 }
```

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.

```
555 // check if character level up or not
556 tama.levelUp();
557 if (getLevelUpTrue() == 1) // if character level up then show dialog message
558 {
559     JOptionPane.showMessageDialog(null, "Congraturation!!! you LEVEL UP!!!!", "level up",
560     JOptionPane.INFORMATION_MESSAGE);
561     Thread.sleep(500);
562     setLevelUpTrue(0); // re-set level change status
563 }
564
565 if (tama.getLevel() > 4) {
566     tama.graduate();
567     JOptionPane.showMessageDialog(null,
568     "Congratulation!!! Your charater graduates from Sungkyunkwan!!!\nGood Job!!!", "Graduate",
569     JOptionPane.INFORMATION_MESSAGE);
570     Thread.sleep(1000);
571     System.exit(1);
572 }
```

```

573
574 // if energy less than 0 then quit the system
575 if (tama.getEnergy() < 0) {
576     tama.expel();
577     JOptionPane.showMessageDialog(null, "Your charater is expelled from Sungkyunkwan..Sorry", "Expel",
578         JOptionPane.INFORMATION_MESSAGE);
579     Thread.sleep(1000);
580     System.exit(1);
581 }
582
583 return 0;
584 }
585

```

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

```

588 @Override
589 // update the GUI here.
590 protected void process(List<String> chunks) {
591     String component = chunks.get(chunks.size() - 1); // get the string
592     String[] contentsList = new String[eventCase];
593     contentsList[chunks.size() - 1] = component; // save one component into the array
594     for (int i = 0; i < chunks.size(); i++) // update array through the loop
595     {
596         all = all + contentsList[i] + "\n";
597     }
598     status.setText(all); // set the text at the text Area
599 }
600
601

```

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

```

602 @Override
603 // this is the case of thread finishes and update GUI here.
604 protected void done() {
605     all = status.getText();
606     all = all + "끝!\n활동을 마치고\n돌아왔어요!\n";
607     status.setText(all); // set the text at the text Area
608     characterLbl.setIcon(new ImageIcon("img\\" + tama.getName() + ".png"));
609
610     // make buttons enable
611     foodBtn.setEnabled(true);
612     sleepBtn.setEnabled(true);
613     playBtn.setEnabled(true);
614     trainBtn.setEnabled(true);
615 }
616 };
617 worker.execute();
618 }

```

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

```
110
111 Character tama = new Character(); //constructor of character;
112
113
134 // Polymorphism
135 if (getCharacter() == 0) {
136     tama = new Girl();
137 } else if (getCharacter() == 1) {
138     tama = new Boy();
139 }
140
```

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

```
403 private class Girl extends Character {
404
405     private String name = "myung";
406
407     public String getName() {
408         return name;
409     }
410
411     private int energy = 30; // re - set energy different from (
412
413     // get set fuction
414     public int getEnergy() {
415         return energy;
416     }
417
418     public void setEnergy(int energy) {
419         this.energy = energy;
420     }
421
443
444 private class Boy extends Character {
445
446     private String name = "yule";
447
448     public String getName() {
449         return name;
450     }
451 }
```

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

```
74      // set girl button
75      girlBtn = new JButton("Myeongnyun");
76      girlBtn.addActionListener(new ActionListener() {
77      public void actionPerformed(ActionEvent e) {
78          setCharacterSex(0); // set character as Myeongnyun
79          dispose(); //close the intro page
80
81          try {
82              FileOutputStream fileObject =new FileOutputStream("data.txt", false);
83
84              PrintWriter x = new PrintWriter(fileObject);
85
86              x.println(getCharacterSex() + "");
87              x.close();
88          } catch (FileNotFoundException e1) {
89              e1.printStackTrace();
90          }
```

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.

```
116      public GameStart() {
117
118          // get the character type through File I/O
119          FileInputStream fileObject;
120          try {
121              fileObject = new FileInputStream("data.txt");
122
123              Scanner x = new Scanner(fileObject);
124
125              while (x.hasNext()) {
126                  String sex = x.nextLine();
127                  character = Integer.parseInt(sex);
128              }
129              x.close();
130          } catch (FileNotFoundException e) {
131              e.printStackTrace();
132          }
```

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.