




GUI 팀 프로젝트

- 주니어 네이버 구현
 - [타자게임] 에서 Thread를 통해 순위 비교 구현
 - [스네이크 게임] 에서 singleton 적용
- 

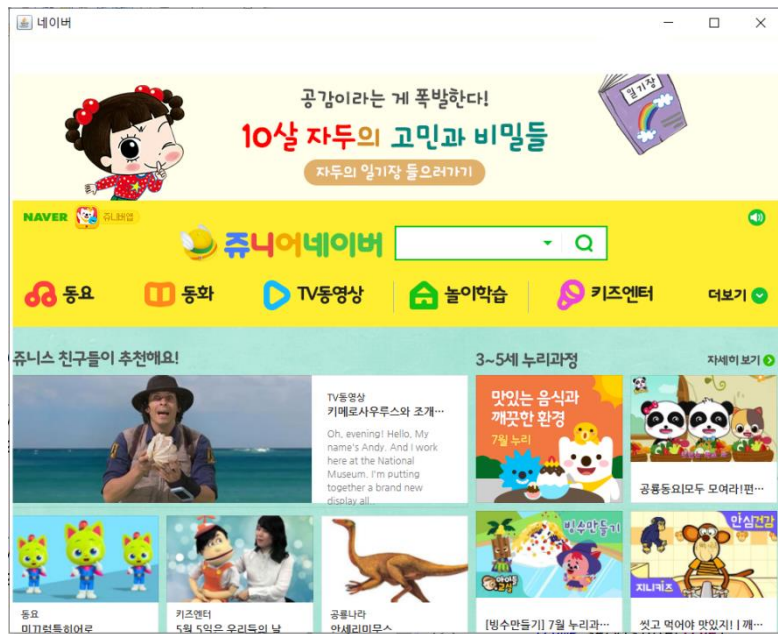
contents

I. 프로젝트 내용

- 01 선정 이유
- 02 역할 및 기여도

II. 프로젝트 설명

- 01 일정
- 02 기능 구현





I. 프로젝트 내용

01 선정 이유

02 역할 및 기여도

I. 프로젝트 내용

01 선정 이유

- 1주일이라는 제한된 시간 내에 프로젝트를 완성해야 했기에 전반적인 '완성도'를 높이는 것을 목표로 정했습니다.
- Java로 게임을 구현해왔던 것들 활용하여 미니게임을 만들고 디자인 측면에서 완성도를 높일 수 있을 것이라고 보았습니다.

I. 프로젝트 내용

02 역할 및 기여도 | back-end



기여도

100%

기술 스펙

사용언어 : Java

주요 내용

'한글과 컴퓨터' 타자 연습과 같이 화면에 보여지는 단어를 빠르게 적고, 총 걸린 시간에 따라 순위가 매겨지도록 하였습니다.

I. 프로젝트 내용

02 역할 및 기여도 | back-end



기여도

50%(게임이 진행되는 것은 구현하였고 이를 미니게임에 붙이는 과정에서는 다른 팀원과 협업하여 진행하였습니다.)

기술 스펙

사용언어 : Java

주요 내용

스네이크를 이동시키면서 사과를 먹으면 꼬리가 길어지고, 벽을 만나거나 자신의 몸을 먹게 되면 게임이 다시 리셋되도록 작성을 하였습니다.

I. 프로젝트 내용

02 역할 및 기여도 | front-end



1 to 50
타자게임
스네이크
게임

역할 및 기여도

100%

실제 게임 화면 배경을 Jpanel에 붙임으로써 실제와 유사하게 만들기 위해 노력했습니다.

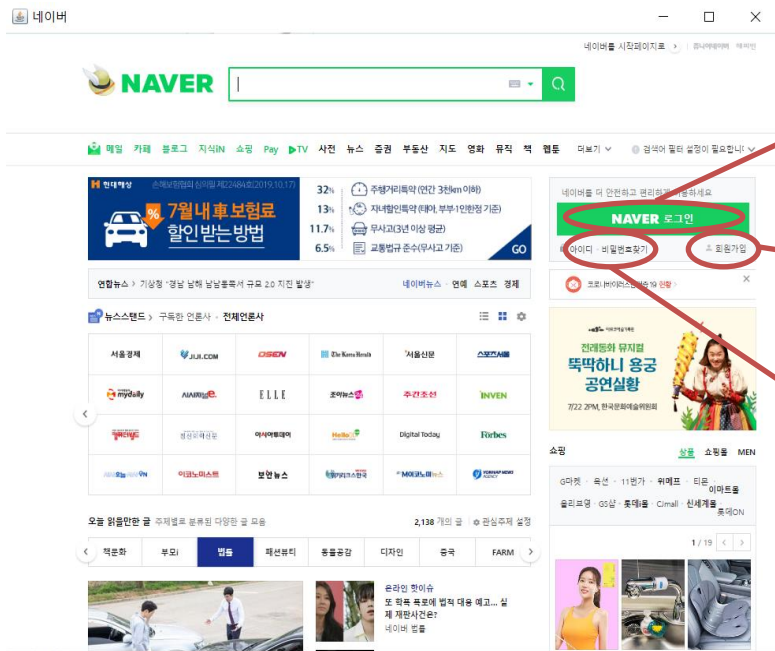
II. 프로젝트 설명

01 일정

	1일차	2일차	3일차	4일차	5일차	6일차	7일차	8일차
주제선정								
구조, 콘텐츠 설계								
미니게임 코드 작성								
미니게임 디자인								
로그인 화면								
메인 화면								
미니게임 랭킹패널								
Merge & Debugging								

II. 프로젝트 설명

02 기능 구현_메인 화면



1. 로그인 기능 구현

2. 회원가입

3. 아이디, 비밀번호 찾기



II. 프로젝트 설명

02 기능 구현_메인 화면

네이버



메인 화면

```
layeredPane.setSize(1000, 800);
```

콘솔 창 = 1000*800으로 고정

```
Title_Panel()
{
    //추가
    JLayeredPane layeredPane = new JLayeredPane();
    layeredPane.setSize(1000, 800);
    layeredPane.setLayout(null);

    try
    {
        img = ImageIO.read(new File("C:\\Users\\duar1\\eclipse-workspace\\miniGame\\src\\gameBoard\\네이버창.png"));
    }
    catch (IOException e)
    {
        JOptionPane.showMessageDialog(null, "이미지 불러오기 실패");
        System.exit(0);
    }

    myPanel panel = new myPanel();
    panel.setSize(1000, 800);

    setLayout(null);

    setBounds(500, 400, 1000, 800);

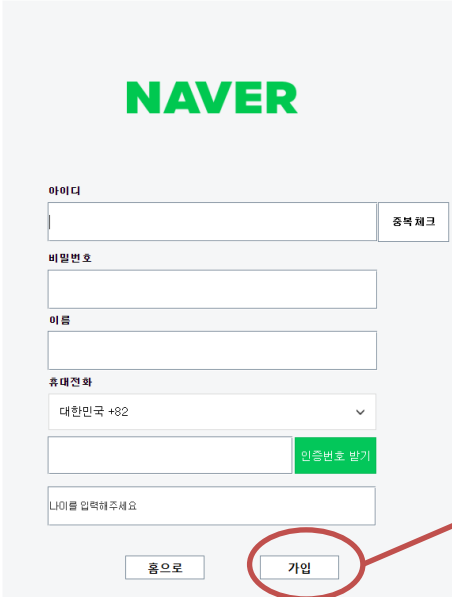
    layeredPane.add(panel);
    add(layeredPane);
    setVisible(true);
}

class myPanel extends JPanel
{
    public void paint(Graphics g)
    {
        super.paintComponent(g);
        g.drawImage(img, 0, 0, null);
    }
}
```

디자인 코드

II. 프로젝트 설명

02 기능 구현_로그인, 회원가입, 중복 검사



A screenshot of the Naver registration form in a web browser window. The form includes fields for ID, password, name, phone number, and email. A red circle highlights the '가입' (Sign Up) button at the bottom right.

네이버

NAVER

아이디

중복 체크

비밀번호

이름

휴대전화

대한민국 +82


인증번호 받기

나이를 입력해주세요

홈으로

가입

회원 가입



A screenshot of the Naver registration form with a duplicate ID error message. The ID field contains 'mega'. A red arrow points from the '가입' button in the previous screenshot to this error message.

네이버

NAVER

아이디

mega

중복 체크

비밀번호

1234

이름

메가아이디

휴대전화

대한민국 +82

0108888

인증번호 받기

25

홈으로

가입

중복 확인 미실행

중복 확인을 해주세요

확인

아이디 중복 검사

II. 프로젝트 설명

02 기능 구현_로그인, 회원가입, 중복 검사

네이버

The image shows the Naver login/signup form. The '아이디' (ID) field contains 'mega'. A modal message box is displayed in the center, stating '아이디 중복 확인' (Check for duplicate ID) and '사용 가능한 아이디입니다' (This is a usable ID). The form includes fields for '비밀번호' (Password), '이름' (Name), '휴대전화' (Mobile phone), and '생년월일' (Date of birth). There are buttons for '로그인' (Login) and '회원가입' (Sign up).

아이디 중복 체크

네이버

The image shows the Naver login/signup form after successful registration. A modal message box is displayed, stating '회원가입 축하드립니다.' (Congratulations on your registration). The form fields are the same as in the previous image, but the '아이디' field now contains 'mega' and the '비밀번호' field is empty. The '로그인' (Login) button is highlighted.

회원 정보 저장

회원 가입 완료

== S A V E ==

말보루/25/0101111/qwer/1234/0
에세제인지업/25/0102222/asdf/1234/0
팔라/25/0103333/zxcv/1234/0
메가아이티/25/0108888/mega/1234/0

ArrayList<UserInfo> userManager = null;

II. 프로젝트 설명

02 기능 구현_로그인, 회원가입, 중복 검사

네이버

아이디 찾기

☒ 회원정보에 등록된 휴대전화로 인증
회원정보에 등록된 휴대전화 번호와 입력한 휴대전화 번호가 같아야, 인증번호를 받을 수 있습니다.

이름

휴대전화 인증번호 받기

인증번호가 오지 않나요 [?] **ID 찾기**

☐ 본인확인 이메일로 인증

비밀번호 찾기

☒ 회원정보에 등록된 휴대전화로 인증
회원정보에 등록된 휴대전화 번호와 입력한 휴대전화 번호가 같아야, 인증번호를 받을 수 있습니다.

이름

휴대전화 인증번호 받기

인증번호가 오지 않나요 [?] 아이디 입력

PW 찾기

처음화면

ID/PW 찾기

네이버

아이디 찾기

☒ 회원정보에 등록된 휴대전화로 인증
회원정보에 등록된 휴대전화 번호와 입력한 휴대전화 번호가 같아야, 인증번호를 받을 수 있습니다.

이름

휴대전화 인증번호 받기

인증번호가 오지 않나요 [?] **ID 찾기**

☐ 본인확인 이메일로 인증

비밀번호 찾기

☒ 회원정보에 등록된 휴대전화로 인증
회원정보에 등록된 휴대전화 번호와 입력한 휴대전화 번호가 같아야, 인증번호를 받을 수 있습니다.

이름

휴대전화 인증번호 받기

인증번호가 오지 않나요 [?] 아이디 입력

PW 찾기

처음화면

아이디 찾기

아이디 : qwer

확인

ID 찾기

II. 프로젝트 설명

02 기능 구현_로그인, 회원가입, 중복 검사

네이버

아이디 찾기

☒ 회원정보에 등록된 휴대전화로 인증

회원정보에 등록된 휴대전화 번호와 입력한 휴대전화 번호가 같아야, 인증번호를 받을 수 있습니다.

이름

휴대전화 인증번호 받기

인증번호가 오지 않나요?

ID 찾기

☐ 본인확인 이메일로 인증

비밀번호 찾기

☒ 회원정보에 등록된 휴대전화로 인증

회원정보에 등록된 휴대전화 번호와 입력한 휴대전화 번호가 같아야, 인증번호를 받을 수 있습니다.

이름

휴대전화 인증번호 받기

인증번호가 오지 않나요?

아이디 입력

PW 찾기

처음화면

ID/PW 찾기

네이버

아이디 찾기

☒ 회원정보에 등록된 휴대전화로 인증

회원정보에 등록된 휴대전화 번호와 입력한 휴대전화 번호가 같아야, 인증번호를 받을 수 있습니다.

이름

휴대전화 인증번호 받기

인증번호가 오지 않나요?

ID 찾기

☐ 본인확인 이메일로 인증

비밀번호 찾기

☒ 회원정보에 등록된 휴대전화로 인증

회원정보에 등록된 휴대전화 번호와 입력한 휴대전화 번호가 같아야, 인증번호를 받을 수 있습니다.

이름

휴대전화 인증번호 받기

인증번호가 오지 않나요?

아이디 입력

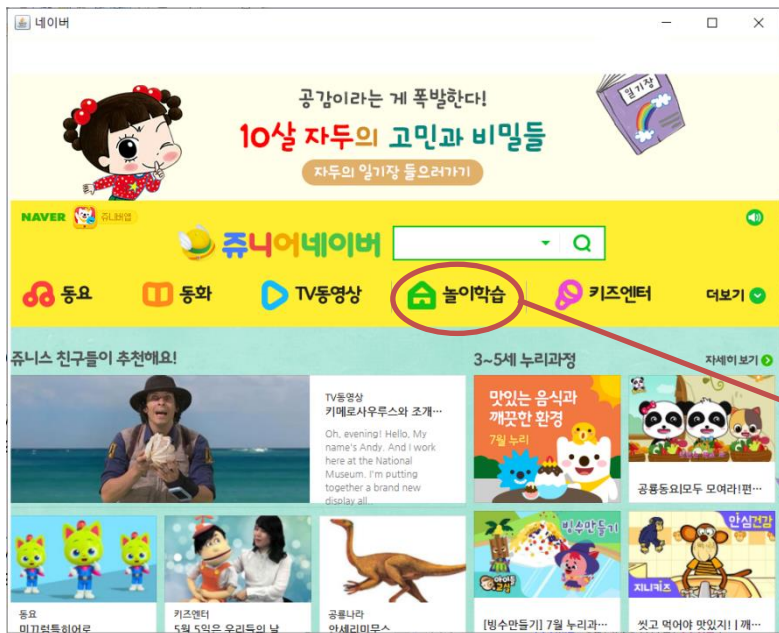
PW 찾기

처음화면

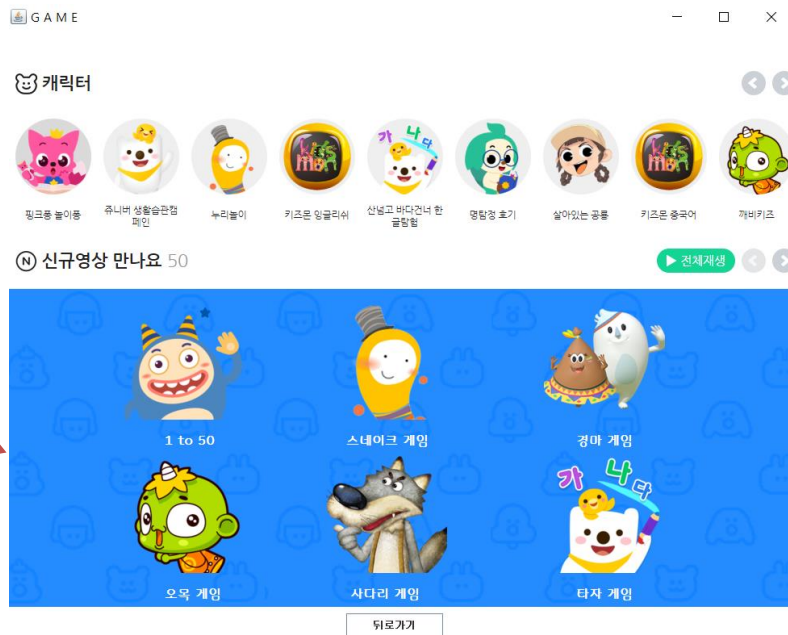
PW 찾기

II. 프로젝트 설명

02 기능 구현_로그인, 회원가입, 중복 검사



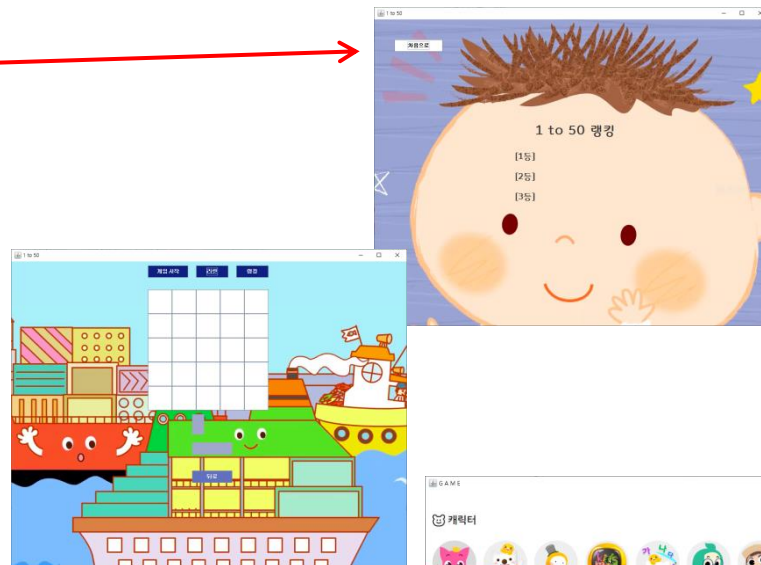
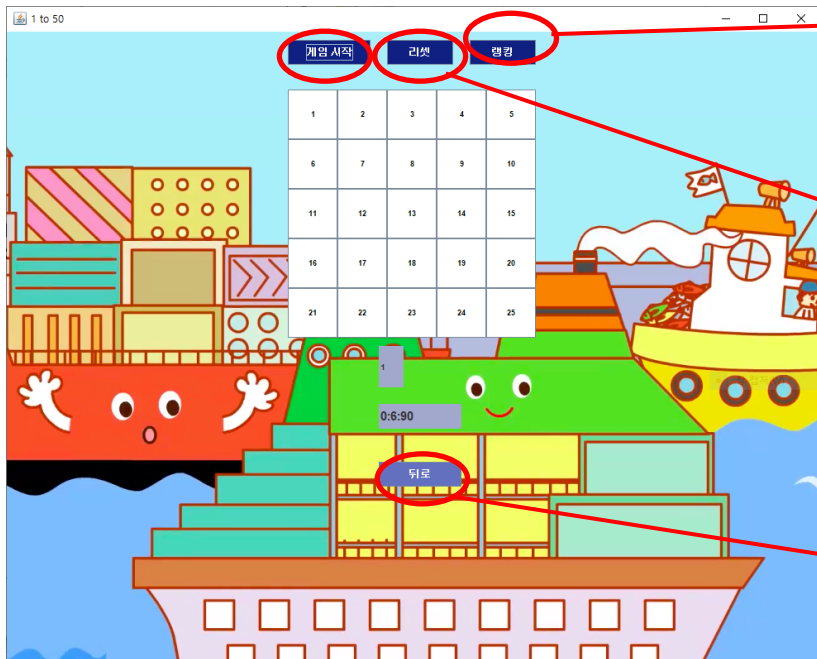
로그인 후 화면



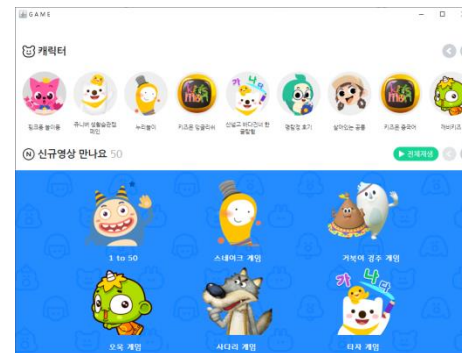
게임 리스트

II. 프로젝트 설명

02 기능 구현_1 TO 50

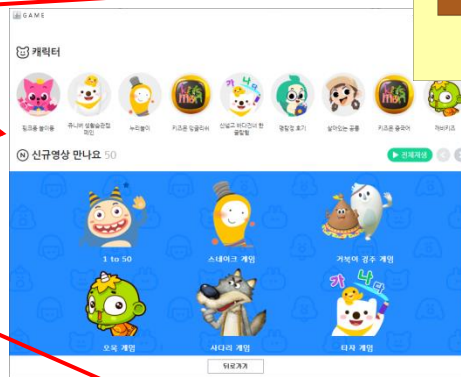
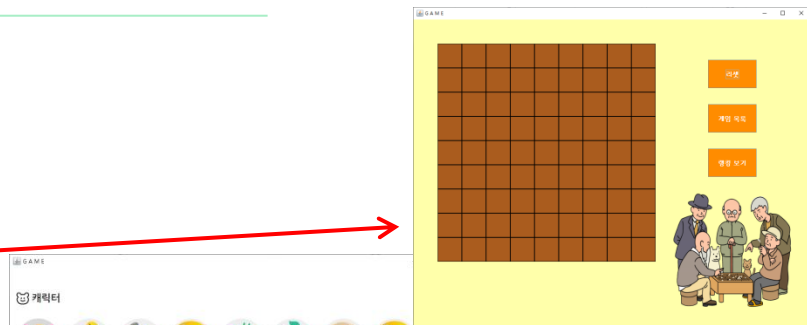
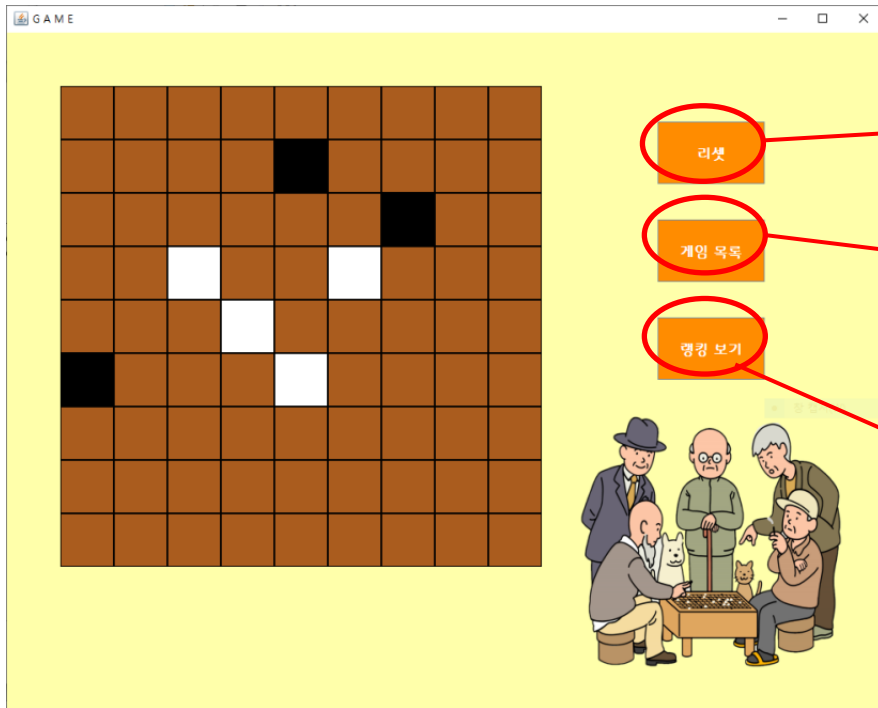


- 게임에 공통적으로 들어가 있는 버튼: 리셋, 랭킹, 뒤로 가기
- 게임마다 다른 이미지를 백그라운드로 사용



II. 프로젝트 설명

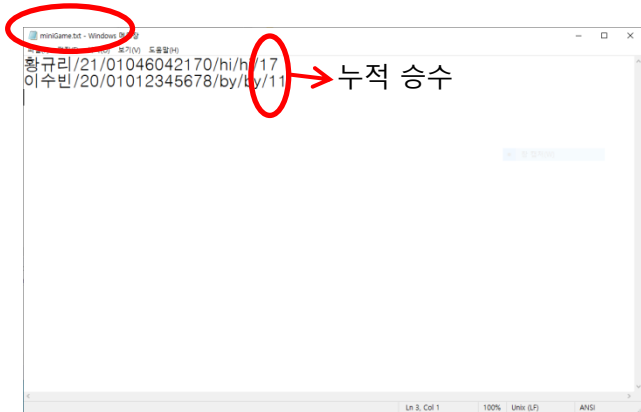
02 기능 구현_오목 게임



II. 프로젝트 설명

02 기능 구현_오목 게임

miniGame.txt: 유저들의 정보가 저장되어 있는 파일



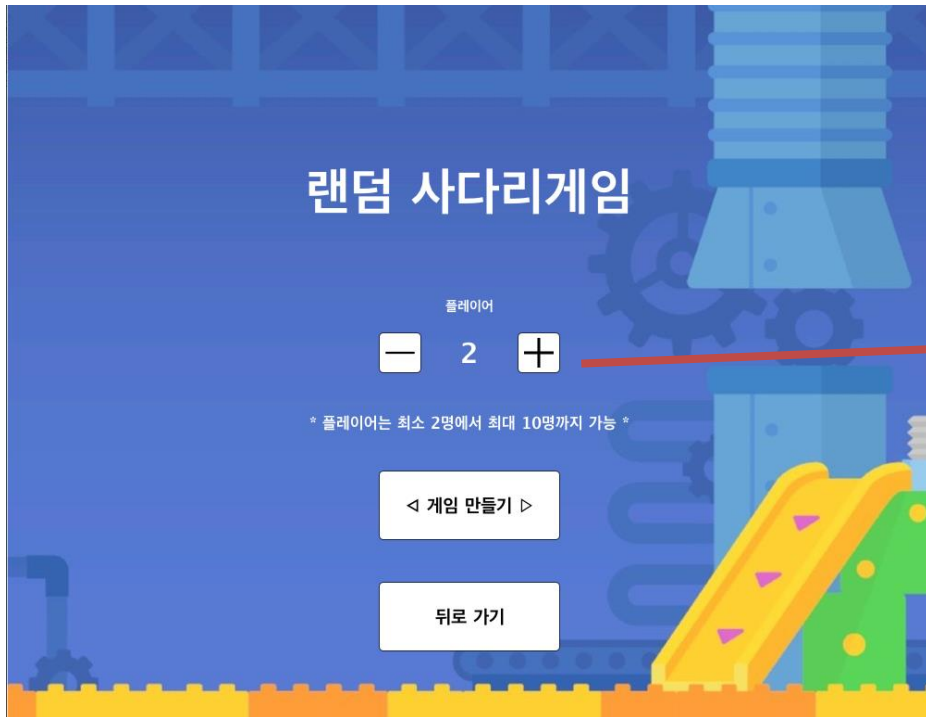
- 오목 게임 랭킹 기준: 누적 승수

```
if(win == 1) {
    JOptionPane.showMessageDialog(null, FileManager.instance.userManager.get(FileManager.LOG).getName()
        + "님 승리!", "오목게임 결과", JOptionPane.WARNING_MESSAGE);
    int newWin = FileManager.instance.userManager.get(FileManager.LOG).getOmokWin();
    newWin++; //로그인 되어있는 유저 아이디의 누적승수 증가
    FileManager.instance.userManager.get(FileManager.LOG).setOmokWin(newWin); //누적 승수 업데이트
    omok_Rank.instance.setInfo(); //오목 게임 랭킹을 정함
    FileManager.instance.saveOmokRankData();
    Update(); //miniGame.txt에 저장되어있는 누적승수 또한 업데이트; 이 메서드 없으면 창을 닫고 재실행 했을 때 누적 승수가 초기화됨
    return;
}
```

- 디자인 코드

II. 프로젝트 설명

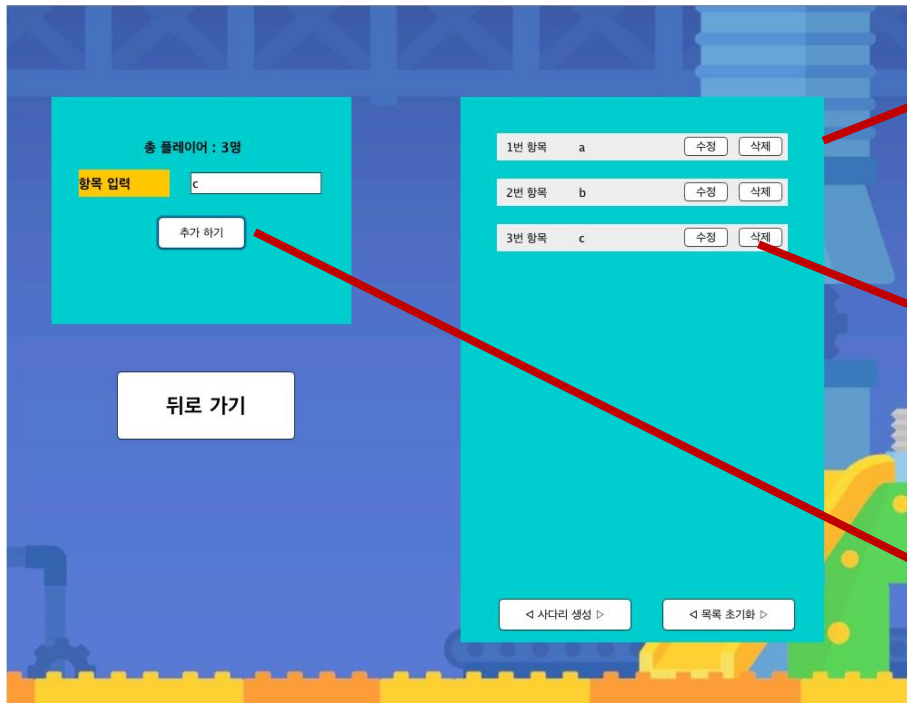
02 기능 구현_랜덤 사다리 게임



- 플레이어 추가, 감소.
- 최소 2명에서 10명까지

II. 프로젝트 설명

02 기능 구현_랜덤 사다리 게임



```
class LadderList extends JPanel implements ActionListener{  
  
    static public Vector<Word> word_list = new Vector<Word>();  
    static public Vector<List> ladderlist = new Vector<>();  
}
```

- LadderList에 Word, List 벡터 선언

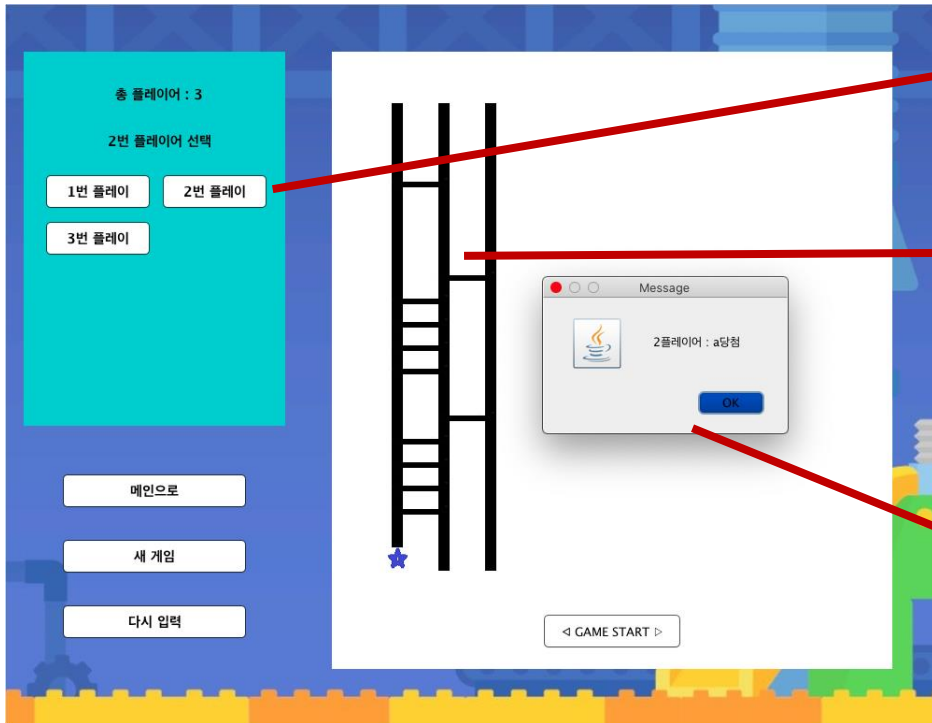
```
List(int index, Word word, LadderList panel) {  
    ladderlist = panel;  
}
```

- List - index 받아서 넘버링, 수정, 삭제 가능

```
public void insert_word(String kor) {  
    Word word = new Word();  
    word.kor = kor;  
    if(word_list.size()<ladder_Game_Panel.player) {  
        word_list.add(word);  
    }  
    print_word_list();  
}
```

II. 프로젝트 설명

02 기능 구현_랜덤 사다리 게임



- 선택한 플레이어 수 만큼 player_Btn 생성
- substring으로 앞 번호 따와서 Cplayer에 저장
- moveLadder 동작

```
if (yIdx == 19) {  
    showLadder();  
    System.out.println(xIdx);  
    run = false;  
}  
try { Thread.sleep(350); } catch (Exception e) { e.printStackTrace(); }  
this.update(this.getGraphics());
```

- 쓰레드로 yIdx가 19될 때까지 update
- xIdx에 해당하는 word_list의 항목과 Cplayer 조합해서 메세지 출력

II. 프로젝트 설명

02 기능 구현_랜덤 사다리 게임

```
String num = player_Btn[i].getText().substring(0,1);  
Cplayer = Integer.parseInt(num);  
print_player_list();
```

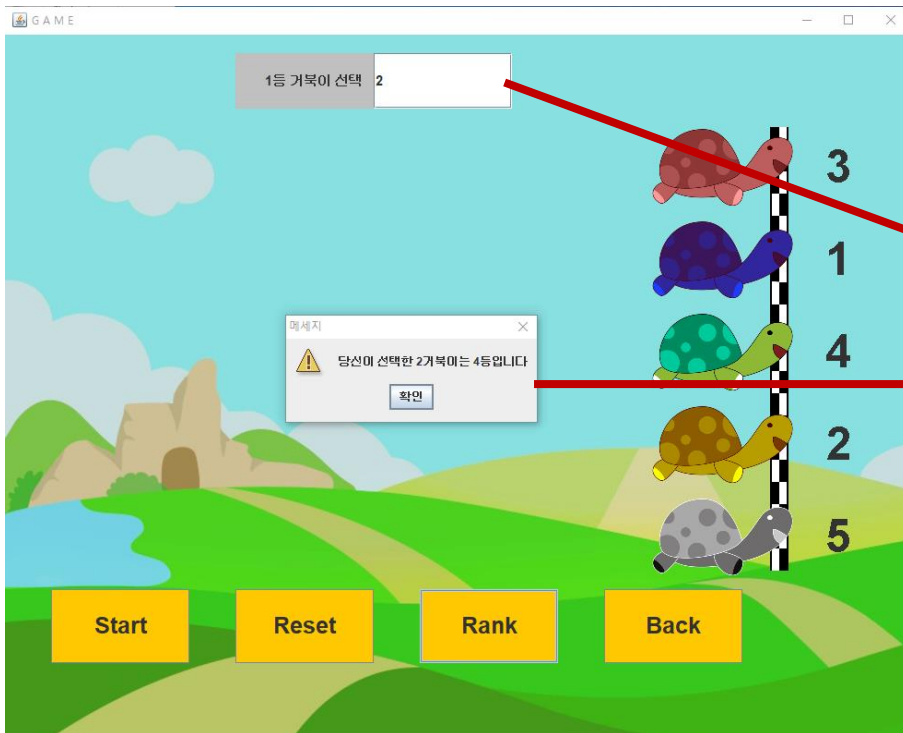
10번째 플레이어 선택 시 이슈



```
void play_btn_click(int i) {  
    if (ladder_Game_Panel.player == 10) {  
        if (i == 9) {  
            String num = "10";  
            Cplayer = Integer.parseInt(num);  
            print_player_list();  
        } else {  
            String num = player_Btn[i].getText().substring(0,1);  
            Cplayer = Integer.parseInt(num);  
            print_player_list();  
        }  
    } else {  
        String num = player_Btn[i].getText().substring(0,1);  
        Cplayer = Integer.parseInt(num);  
        print_player_list();  
    }  
}
```

II. 프로젝트 설명

02 기능 구현_거북이 경주 게임



1. 게임 시작 전에 1등 거북이를 먼저 선택

2. 게임이 끝나면 선택한 거북이의 결과 출력

II. 프로젝트 설명

02 기능 구현_거북이 경주 게임

```
if(check_rank[i] == num_select)
{
    idx = i;
}
```

i : 거북이 등수

num_select : 선택한 거북이

- 선택한 거북이의 등수에 따라 매기는 점수가 차등된다.
- 계정 별로 점수 누적하여 순위 업데이트

```
else
{
    // 등수에 따라 부여 받는 점수가 다르다
    if(idx == 1)
    {
        score = score + 80;
    }
    else if(idx == 2)
    {
        score = score + 60;
    }
    else if(idx == 3)
    {
        score = score + 40;
    }
    else if(idx == 4)
    {
        score = score + 20;
    }
}
```


II. 프로젝트 설명

02 기능 구현_거북이 경주 게임

1. 사용자 정보 저장 시 String, int 등 형 변환에 각별히 주의를 기울였음
2. Thread.sleep 메서드를 활용하여 이미지만으로 동영상과 동일한 콘텐츠를 만들 수 있다는 점 주목
3. 랭킹 산정 시 compareTo 적용
 - 현재 사용자의 기록이 3등보다 작다면 변동 없음
 - 3등보다 크다면 기존 3등을 현 사용자 기록으로 대체
 - 2등, 1등과 비교해 순차적으로 순위 갱신

```
if(rank[2][1].compareTo(score) < 0)
{
    rank[2][0] = id;
    rank[2][1] = score;
}
```

```
sortRank();
```



```
public void sortRank()
{
```

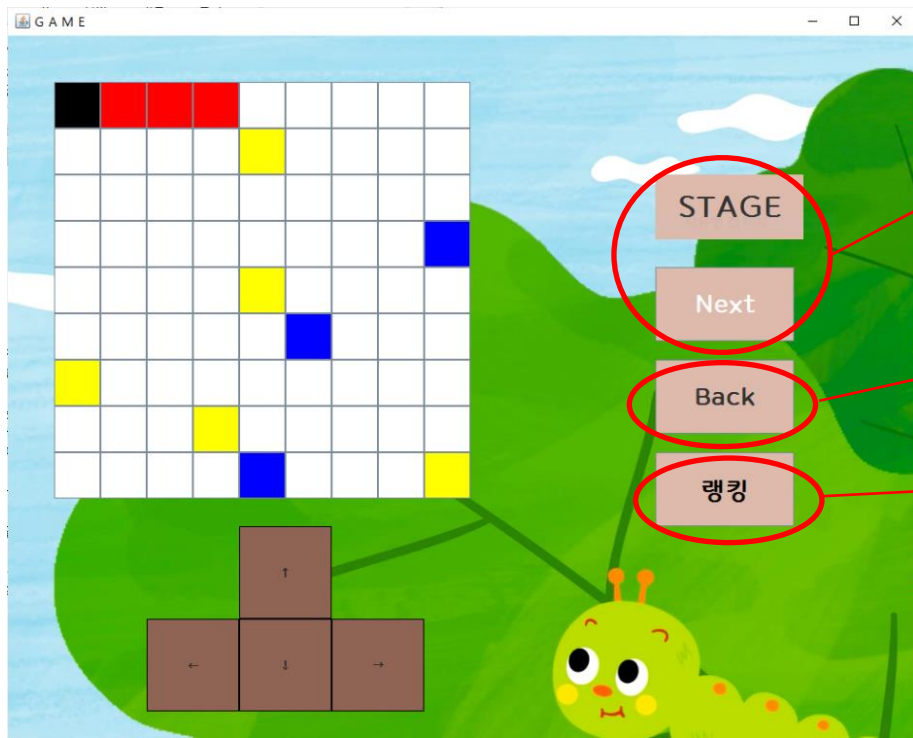
```
    if(rank[i][1].compareTo(rank[j][1]) < 0)
    {
        String tempId = rank[i][0];
        String tempScore = rank[i][1];

        rank[i][0] = rank[j][0];
        rank[i][1] = rank[j][1];

        rank[j][0] = tempId;
        rank[j][1] = tempScore;
    }
}
```

II. 프로젝트 설명

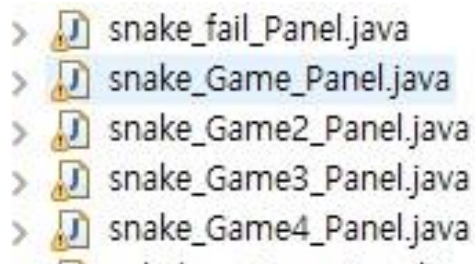
02 기능 구현_스네이크 게임



II. 프로젝트 설명

02 기능 구현_스네이크 게임

- stage, next

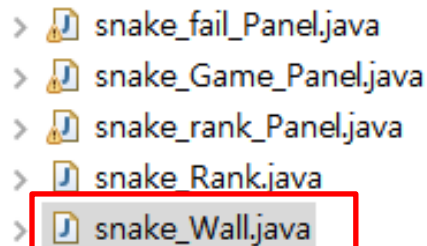


[변경 전]

- stage_Game_panel을 stage 개수 만큼 생성

문제점

- stage 개수의 한계가 존재
- 중복된 코드, 클래스의 생성



[변경 후]

- snake_wall를 싱글톤으로 작성을 하고, get / set 메서드를 통해 wall의 개수를 가져오도록 함.
- stage의 개수 증가에 한계가 없어짐.

II. 프로젝트 설명

02 기능 구현_스네이크 게임

- stage, next

```
snake_Wall.java ✕
1 package gameBoard;
2
3 public class snake_Wall {
4     int wall = 3;
5
6     public static snake_Wall instance = new snake_Wall();
7
8     public int getWall()
9     {
10         return wall;
11     }
12
13     public void setWall(int wall)
14     {
15         this.wall = wall;
16     }
17 }
```

II. 프로젝트 설명

02 기능 구현_스네이크 게임

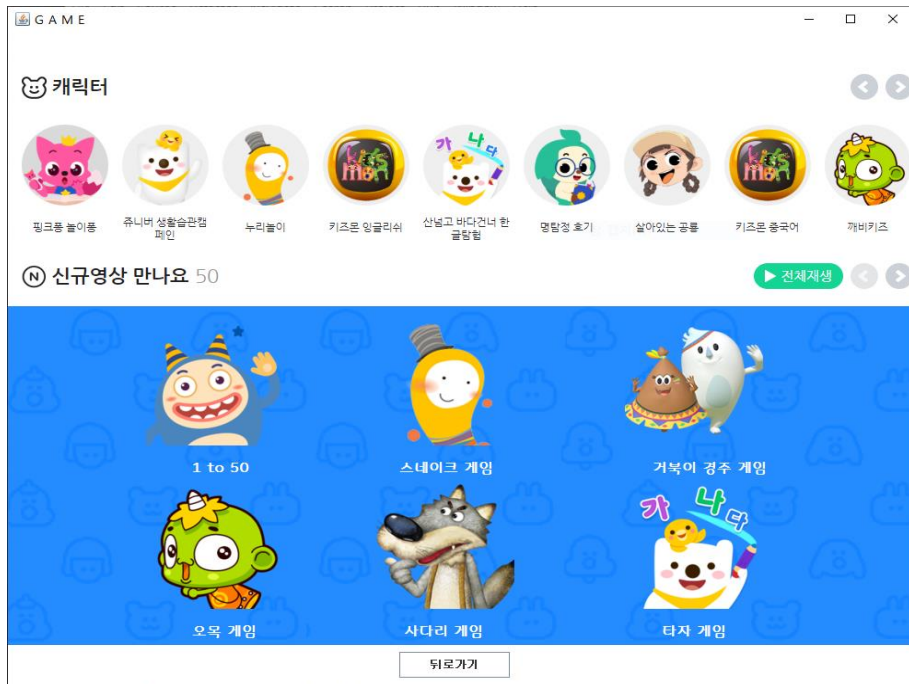
- stage, next

```
if(apple == 0)
{
    JOptionPane.showMessageDialog(null, "클리어!!", "메세지", JOptionPane.WARNING_MESSAGE);
    snake_Wall.instance.wall++;
}
if(crush == 1)
{
    JOptionPane.showMessageDialog(null, "실패하였습니다.", "메세지", JOptionPane.WARNING_MESSAGE);
    _WordGame.frame.setContentPane(new snake_fail_Panel());
    _WordGame.frame.revalidate();
    snake_Wall.instance.setWall(3);
}
```

II. 프로젝트 설명

02 기능 구현_스네이크 게임

- 뒤로 가기



- Game_panel로 넘어간다.
- 뒤로 가기의 기능은 다른 게임에서도 유사

II. 프로젝트 설명

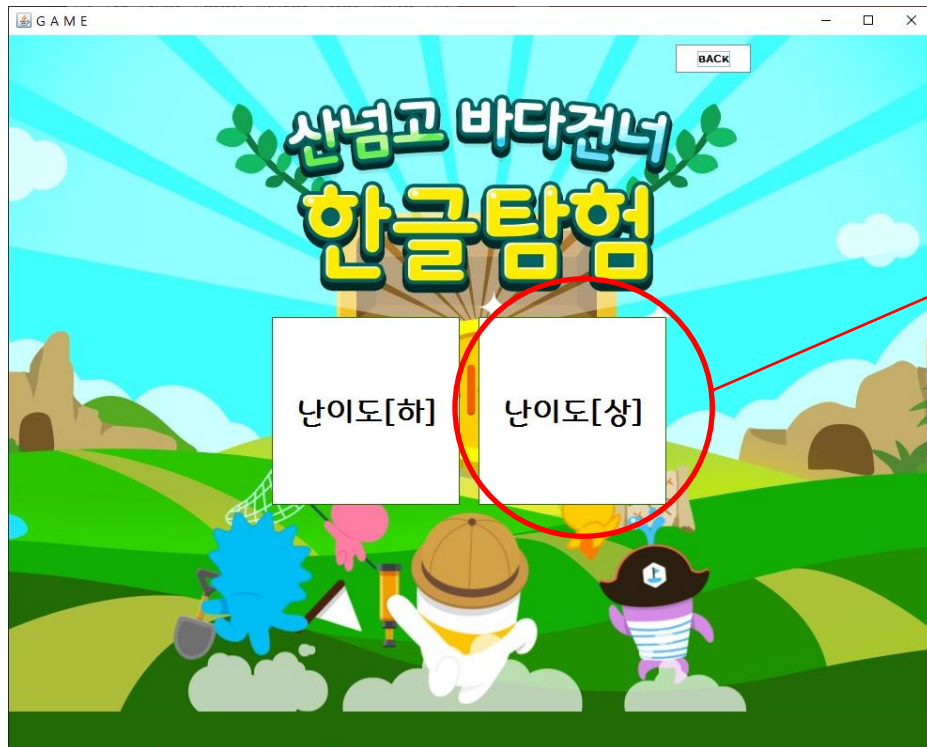
02 기능 구현_스네이크 게임



- 순위 기준 : 올라간 stage 수
- 중복 검사를 하지 않음

II. 프로젝트 설명

02 기능 구현_타자 게임



1. 난이도 결정

II. 프로젝트 설명

02 기능 구현_타자 게임



[리셋]
리셋을 먼저 누르게 될 경우 오류가
발생하여 try, catch를 사용하여
해결함

[랭킹]
게임 진행 시간을 비교하여 랭킹을
만들었음

감사합니다.

지원자 최지완 드림