**HHH**

**Algorithm Term Project – Team 5**

**김현민, 이라연, 이주원**

**[Algorithm – Strategy #1]**

Check that the seating arrangement is entered correctly.

Check that the seating person is far enough away from the entered seating table.

(Criterion: Manhattan distance > 2 is safe, Manhattan distance = 2 is safe with a partition between seats.)

package part;

import java.util.LinkedList;

import java.util.Queue;

import java.util.\*;

class Solution {

    final static int row = 3;

    final static int col = 6;

    public static void main(String[] args) {

        // TODO Auto-generated method stub

        Scanner kb = new Scanner(System.in);

        String[] arr = new String[row];

        // input the table information

        // 유저로부터 테이블 상태 입력받기

        for (int i = 0; i < row; i++) {

            arr[i] = kb.nextLine();

        }

        System.out.println();

        int ans;

        // solution function

        ans = solution(arr);

        System.out.print(ans + " ");

    }

    public static int solution(String[] places) {

        int answer;

        String[] p = places;

        // isOk 변수

        // 거리두기가 지켜지고 있는지 확인하기 위한 변수

        boolean isOk = true;

        // 모든 자리에서 bfs 실행

        // 사람이 앉아있는 자리 중, 한 자리라도 거리두기가 지켜지지 않으면 false 반환

        for (int r = 0; r < row && isOk; r++) {

            for (int c = 0; c < col && isOk; c++) {

                if (p[r].charAt(c) == 'P') {

                    if (!bfs(r, c, p))

                        isOk = false;

                }

            }

        }

        // 거리두기가 지켜지면 1, 지켜지지 않으면 0 반환

        answer = isOk ? 1 : 0;

        return answer;

    }

    // 각 자리에서 거리두기를 확인하기 위한 bfs 함수

    private static boolean bfs(int r, int c, String[] p) {

        int dr[] = { -1, 1, 0, 0 };

        int dc[] = { 0, 0, -1, 1 };

        // 거리두기 비교하기 위한 queue

        Queue<int[]> queue = new LinkedList<int[]>();

        queue.offer(new int[] { r, c });

        while (!queue.isEmpty()) {

            int[] position = queue.poll();

            // i가 0일 경우 왼쪽으로 1칸

            // i가 1일 경우 오른쪽으로 1칸

            // i가 2일 경우 위로 1칸

            // i가 3일 경우 아래로 1칸

            // for loop를 통해 맨하튼거리 = 1인 자리 비교

            for (int i = 0; i < 4; i++) {

                int nr = position[0] + dr[i];

                int nc = position[1] + dc[i];

                // 가장자리인 경우 빈자리, 거리두기 확인할 필요 없음

                // 처음 시작점 P는 탐색 제외 (nr == r, nc == c)인 부분

                if (nr < 0 || nc < 0 || nr >= row || nc >= col || (nr == r && nc == c))

                    continue;

                // 각 자리로부터 맨하튼 거리 d

                int d = Math.abs(nr - r) + Math.abs(nc - c);

                // 파티션이 있는 경우 : 탐색 진행 X 더 이상 찾을 필요 없음

                // 맨하튼거리가 2 이하이면서 다른 사람이 있는 경우 (P), false 반환

                if (p[nr].charAt(nc) == 'P' && d <= 2)

                    return false;

                // 맨하튼 거리가 2 미만이면서 빈자리인 경우 (O) 다음 탐색

                // 다음 탐색을 할 경우 맨하튼 거리가 1 증가하므로 d < 2인 칸만 탐색

                else if (p[nr].charAt(nc) == 'O' && d < 2)

                    queue.offer(new int[] { nr, nc });

            }

        }

        return true;

    }

}

**[Algorithm – Strategy #2]**

Going further from Strategy 1, get seating charts and people's placements, and find out where partitions are needed.

package p2;

import java.util.LinkedList;

import java.util.Queue;

import java.util.\*;

class Solution {

    final static int row = 3;

    final static int col = 5;

    public static int[][] ptarr = new int[row][col];

    public static void main(String[] args) {

        // TODO Auto-generated method stub

        Scanner kb = new Scanner(System.in);

        String[] arr = new String[row];

        // 파티션이 필요한 자리를 저장하기 위한 string 배열

        for (int i = 0; i < row; i++) {

            for (int j = 0; j < col; j++) {

                ptarr[i][j] = 0;

            }

        }

        // input the table information

        // 유저로부터 테이블 상태 입력받기

        for (int i = 0; i < row; i++) {

            arr[i] = kb.nextLine();

        }

        System.out.println();

        int ans;

        ans = solution(arr);

        // 함수를 통해 얻은 결과 출력

        // 파티션이 필요한 자리 = 1, 그 외의 자리는 0

        for (int i = 0; i < row; i++) {

            for (int j = 0; j < col; j++) {

                System.out.print(ptarr[i][j] + " ");

            }

            System.out.println();

        }

    }

    public static int solution(String[] places) {

        int answer;

        String[] p = places;

        // isOk 변수

        // 거리두기가 지켜지고 있는지 확인하기 위한 변수

        boolean isOk = true;

        // 모든 자리에서 bfs 실행

        // 사람이 앉아있는 자리 중, 한 자리라도 거리두기가 지켜지지 않으면 false 반환

        for (int r = 0; r < row && isOk; r++) {

            for (int c = 0; c < col && isOk; c++) {

                if (p[r].charAt(c) == 'P') {

                    if (!bfs(r, c, p))

                        isOk = false;

                }

            }

        }

        // 거리두기가 지켜지면 1, 지켜지지 않으면 0 반환

        answer = isOk ? 1 : 0;

        return answer;

    }

    // 각 자리에서 거리두기를 확인하기 위한 bfs 함수

    private static boolean bfs(int r, int c, String[] p) {

        int dr[] = { -1, 1, 0, 0 };

        int dc[] = { 0, 0, -1, 1 };

        // 거리두기 비교하기 위한 queue

        Queue<int[]> queue = new LinkedList<int[]>();

        queue.offer(new int[] { r, c });

        while (!queue.isEmpty()) {

            int[] position = queue.poll();

            // i가 0일 경우 위로 1칸

            // i가 1일 경우 아래로 1칸

            // i가 2일 경우 왼쪽으로 1칸

            // i가 3일 경우 오른쪽으로 1칸

            // for loop를 통해 맨하튼거리 = 1인 자리 비교

            for (int i = 0; i < 4; i++) {

                int nr = position[0] + dr[i];

                int nc = position[1] + dc[i];

                // 가장자리인 경우 빈자리, 거리두기 확인할 필요 없음

                // 처음 시작점 P는 탐색 제외 (nr == r, nc == c)인 부분

                if (nr < 0 || nc < 0 || nr >= row || nc >= col || (nr == r && nc == c))

                    continue;

                // 각 자리로부터 맨하튼 거리 d

                int d = Math.abs(nr - r) + Math.abs(nc - c);

                // 파티션이 있는 경우 : 탐색 진행 X 더 이상 찾을 필요 없음

                // 맨하튼거리가 2 이하이면서 다른 사람이 있는 경우(P) 원래는 false 반환하고 함수 종료했지만

                // 나머지 자리들에서 파티션이 필요한 위치를 저장하기 위해 반환하지 않음

                if (p[nr].charAt(nc) == 'P' && d <= 2) {

                    // i = 0인 경우, 위로 한칸 이동한 것이므로 1칸 아래에 파티션 필요

                    if (i == 0) {

                        ptarr[nr + 1][nc] = 1;

                    }

                    // i = 1인 경우, 아래로 한칸 이동한 것이므로 1칸 위에 파티션 필요

                    else if (i == 1) {

                        ptarr[nr - 1][nc] = 1;

                    }

                    // i = 2인 경우, 왼쪽으로 한칸 이동한 것이므로 1칸 오른쪽에 파티션 필요

                    else if (i == 2) {

                        ptarr[nr][nc + 1] = 1;

                    }

                    // i = 3인 경우, 오른쪽으로 한칸 이동한 것이므로 1칸 왼쪽에 파티션 필요

                    else if (i == 3) {

                        ptarr[nr][nc - 1] = 1;

                    }

                    // return false;

                }

                // 맨하튼 거리가 2 미만이면서 빈자리인 경우 (O) 다음 탐색

                // 다음 탐색을 할 경우 맨하튼 거리가 1 증가하므로 d < 2인 칸만 탐색

                else if (p[nr].charAt(nc) == 'O' && d < 2)

                    queue.offer(new int[] { nr, nc });

            }

        }

        return true;

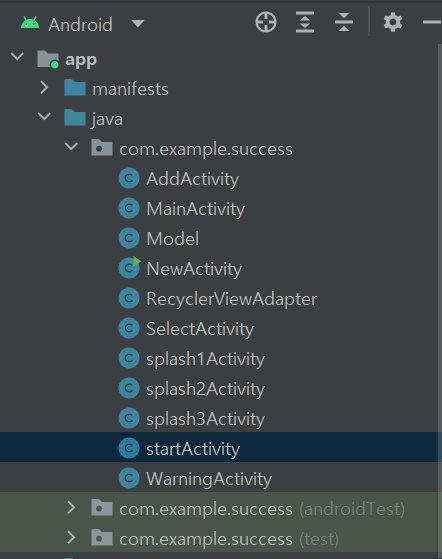
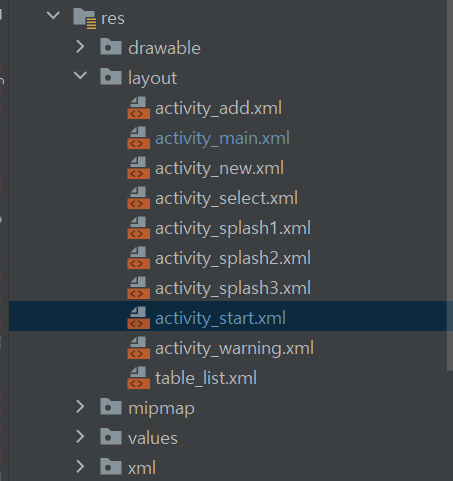
    }

}

**Android Studio**

[Execution Guide]

…

startActivity and activity\_start compose one screen of the application.

The activity file is a java class that contains the functions of how the app will operate on the screen,

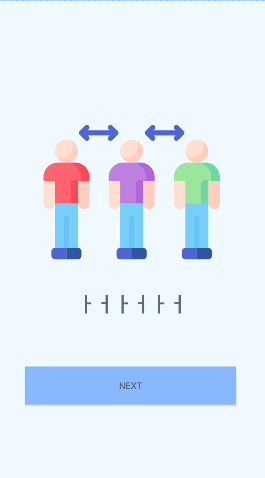
and the xml file is a file that implements the layout. Activity and xml always work as a pair.

**startActivity & activity\_start**

[activity\_start.xml]

<?xml version="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 android:background="#F0F9FF">  
  
  
 <ImageView  
 android:id="@+id/imageView0"  
 android:layout\_width="270dp"  
 android:layout\_height="268.49dp"  
 android:layout\_marginTop="160dp"  
 android:src="@drawable/queue"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.496"  
 app:layout\_constraintHorizontal\_chainStyle="spread"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent" />  
  
 <TextView  
 android:id="@+id/some\_id"  
 android:layout\_width="271dp"  
 android:layout\_height="54dp"  
 android:layout\_marginTop="438dp"  
 android:gravity="center"  
 android:text="ㅏㅓㅏㅓㅏㅓ"  
 android:textColor="#566C81"  
 android:textSize="30sp"  
 android:textStyle="bold"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_chainStyle="spread"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent" />  
  
 <Button  
 android:id="@+id/next"  
 android:layout\_width="321dp"  
 android:layout\_height="58dp"  
 android:layout\_marginLeft="13dp"  
 android:layout\_marginTop="560dp"  
 android:background="#89B8FF"  
 android:textColor="#566C81"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.388"  
 app:layout\_constraintHorizontal\_chainStyle="spread"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 android:text="NEXT"/>  
  
</androidx.constraintlayout.widget.ConstraintLayout>

In order to implement the start screen, an ImageView to contain the app's logo, a TextView to indicate the app's name, and a Button to insert a button that is an interface to move to the next screen are inserted in the xml file in order. The logo of the app uses a png file, and the png file needed is put in the res/drawble file.



The layout of the screen is implemented as shown in the figure.

Write a function that needs to be operated in startActivity to go to the next screen when the NEXT button is pressed..

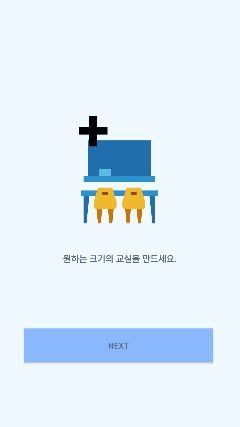
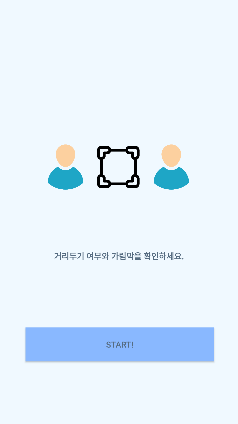
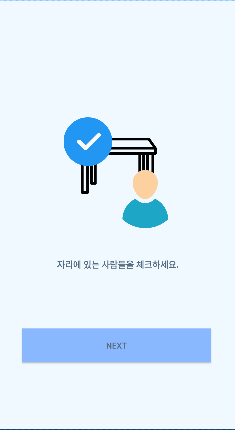
[startActivity]

package com.example.success;  
  
  
import android.content.Intent;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.Button;  
  
import androidx.annotation.Nullable;  
import androidx.appcompat.app.AppCompatActivity;  
  
public class startActivity extends AppCompatActivity {  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_start*);  
  
 Button imageButton = (Button) findViewById(R.id.*next*);  
 imageButton.setOnClickListener(new View.OnClickListener() {  
  
 @Override  
 public void onClick(View view) {  
 Intent intent = new Intent(getApplicationContext(), splash1Activity.class);  
 startActivity(intent);  
 }  
 });  
 }  
}

The id of the Button containing the text NEXT in the xml file is next. In order to give this button a function, create a function to operate when the button is pressed using the button's id in the class. When the button is pressed, it moves to splash1Activity.

[Same action as start for splash1,2,3]

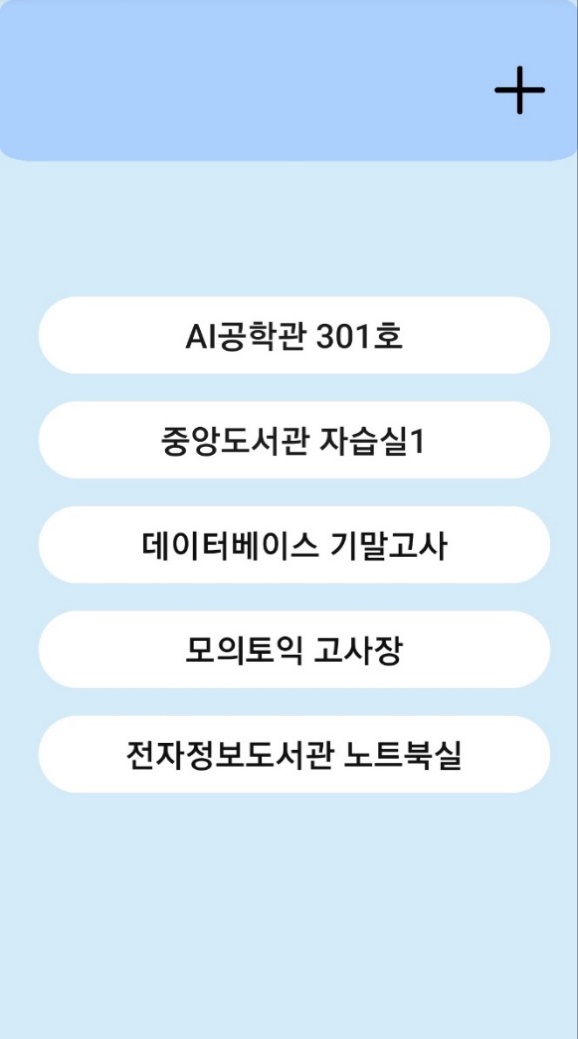
Splash1 Splash2 Splash3

[activity\_main.xml]

<?xml version="1.0" encoding="utf-8"?>  
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 android:id="@+id/app\_main"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="fill\_parent"  
 android:background="@drawable/app\_add"  
 android:clipToOutline="true"  
 android:visibility="visible"  
 app:layout\_constraintGuide\_percent="0.3">  
  
 <View  
 android:id="@+id/rectangle\_6"  
 android:layout\_width="match\_parent"  
 android:layout\_height="102dp"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentTop="true"  
 android:layout\_marginLeft="0dp"  
 android:layout\_marginTop="0dp"  
 android:background="@drawable/rectangle\_6" />  
  
 <Button  
 android:id="@+id/btn\_add"  
 android:layout\_width="42dp"  
 android:layout\_height="40dp"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentTop="true"  
 android:layout\_marginLeft="303dp"  
 android:layout\_marginTop="38dp"  
 android:background="@drawable/plusclassroom" />  
  
 <Button  
 android:id="@+id/newClass"  
 android:layout\_width="321dp"  
 android:layout\_height="50dp"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentTop="true"  
 android:layout\_marginLeft="23dp"  
 android:layout\_marginTop="185dp"  
 android:background="@drawable/input\_border"  
 android:onClick="gotoselect"  
 android:text="AI공학관 301호"  
 android:textColor="#1C1B1B"  
 android:visibility="visible" />  
  
 <Button  
 android:id="@+id/newClass2"  
 android:layout\_width="321dp"  
 android:layout\_height="50dp"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentTop="true"  
 android:layout\_marginLeft="23dp"  
 android:layout\_marginTop="250dp"  
 android:background="@drawable/input\_border"  
 android:text="중앙도서관 자습실1"  
 android:onClick="gotoselect"  
 android:textColor="#1C1B1B"  
 android:visibility="visible" />  
  
 <Button  
 android:id="@+id/newClass3"  
 android:layout\_width="321dp"  
 android:layout\_height="50dp"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentTop="true"  
 android:layout\_marginLeft="23dp"  
 android:layout\_marginTop="315dp"  
 android:background="@drawable/input\_border"  
 android:text="데이터베이스 기말고사"  
 android:textColor="#121111"  
 android:onClick="gotoselect"  
 android:visibility="visible" />  
  
 <Button  
 android:id="@+id/newClass4"  
 android:layout\_width="321dp"  
 android:layout\_height="50dp"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentTop="true"  
 android:layout\_marginLeft="23dp"  
 android:layout\_marginTop="380dp"  
 android:background="@drawable/input\_border"  
 android:text="모의토익 고사장"  
 android:textColor="#121111"  
 android:onClick="gotoselect"  
 android:visibility="visible" />  
  
 <Button  
 android:id="@+id/newClass5"  
 android:layout\_width="321dp"  
 android:layout\_height="50dp"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentTop="true"  
 android:layout\_marginLeft="23dp"  
 android:layout\_marginTop="445dp"  
 android:background="@drawable/input\_border"  
 android:text="전자정보도서관 노트북실"  
 android:textColor="#121111"  
 android:onClick="gotoselect"  
 android:visibility="visible" />  
  
 <Button  
 android:id="@+id/newClass6"  
 android:layout\_width="321dp"  
 android:layout\_height="50dp"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentTop="true"  
 android:layout\_marginLeft="23dp"  
 android:layout\_marginTop="510dp"  
 android:background="@drawable/input\_border"  
 android:textColor="#121111"  
 android:onClick="gotoselect"  
 android:visibility="invisible" />  
  
</RelativeLayout>

This is the main screen layout for creating a classroom.



In order to give a function to the layout screen created as shown in the picture, implement a function that will operate when the + button is pressed in MainActivity.

[MainActivity]

package com.example.success;  
  
import androidx.activity.result.ActivityResultLauncher;  
import androidx.activity.result.contract.ActivityResultContracts;  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.core.app.ActivityCompat;  
  
import android.content.Intent;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.Button;  
  
public class MainActivity extends AppCompatActivity {  
  
 Button newClass;  
 String[] numbers;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
  
 newClass = findViewById(R.id.*newClass6*);  
 numbers = new String[]{};  
  
  
 findViewById(R.id.*btn\_add*).setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 Intent intent = new Intent(MainActivity.this,AddActivity.class);  
 getAddActivityResult.launch(intent);  
 }  
 });  
  
  
  
 }  
  
 private final ActivityResultLauncher<Intent> getAddActivityResult = registerForActivityResult(  
 new ActivityResultContracts.StartActivityForResult(),  
 result -> {  
 if(result.getResultCode() == *RESULT\_OK*){  
  
 newClass.setText(result.getData().getStringExtra("className"));  
 newClass.setVisibility(newClass.*VISIBLE*);  
 numbers = result.getData().getStringArrayExtra("numbers");  
  
 }  
 }  
 );  
  
 public void gotoselect(View view){  
 Intent intent2 = new Intent(MainActivity.this,SelectActivity.class);  
 //intent2.putExtra("numbers",numbers);  
 startActivity(intent2);  
 finish();  
 }  
  
  
}

If you press the + button, it is connected to AddActivity where you enter information about a new classroom.

[activity\_add.xml]

<?xml version="1.0" encoding="utf-8"?>  
<RelativeLayout  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 android:id="@+id/app\_add"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="fill\_parent"  
 android:clipToOutline="true"  
 android:background="@drawable/app\_add"  
  
 >  
  
 <View  
 android:id="@+id/rectangle\_6"  
 android:layout\_width="match\_parent"  
 android:layout\_height="203dp"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentTop="true"  
 android:layout\_marginLeft="0dp"  
 android:layout\_marginTop="0dp"  
 android:layout\_marginEnd="0dp"  
 android:background="@drawable/rectangle\_6"  
 android:gravity="center"/>  
  
  
 <ImageView  
 android:id="@+id/classroom\_1"  
 android:layout\_width="130dp"  
 android:layout\_height="130dp"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentTop="true"  
 android:layout\_marginLeft="115dp"  
 android:layout\_marginTop="45dp"  
 android:background="@drawable/classroom" />  
  
  
 <TextView  
 android:id="@+id/textbox"  
 android:layout\_width="295dp"  
 android:layout\_height="64dp"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentTop="true"  
 android:layout\_marginLeft="20dp"  
 android:layout\_marginTop="270dp"  
 android:gravity="top"  
 android:text="@string/className"  
 android:textAppearance="@style/className" />  
  
  
  
  
 <TextView  
 android:id="@+id/x\_"  
 android:layout\_width="295dp"  
 android:layout\_height="64dp"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentTop="true"  
 android:layout\_marginLeft="23dp"  
 android:layout\_marginTop="380dp"  
 android:gravity="top"  
 android:text="@string/array"  
 android:textAppearance="@style/array" />  
  
 <EditText  
 android:id="@+id/firstNum"  
 android:layout\_width="80dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="50dp"  
 android:layout\_marginTop="441dp"  
 android:autofillHints=""  
 android:background="@drawable/search\_box"  
 android:ems="10"  
 android:inputType="text"  
 android:minHeight="40dp"  
 android:paddingStart="10dp"  
 android:paddingEnd="10dp"  
  
 android:textAlignment="center"  
 android:textColor="#FFFFFF"  
 android:textColorHint="#AAAAAA" />  
  
 <EditText  
 android:id="@+id/secondNum"  
 android:layout\_width="80dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="215dp"  
 android:layout\_marginTop="441dp"  
 android:autofillHints=""  
 android:background="@drawable/search\_box"  
 android:ems="10"  
 android:inputType="text"  
 android:minHeight="40dp"  
 android:paddingStart="10dp"  
 android:paddingEnd="10dp"  
  
 android:textAlignment="center"  
 android:textColor="#FFFFFF"  
 android:textColorHint="#AAAAAA" />  
  
  
  
  
  
 <TextView  
 android:id="@+id/x"  
 android:layout\_width="39dp"  
 android:layout\_height="60dp"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentTop="true"  
 android:layout\_marginLeft="160dp"  
 android:layout\_marginTop="417dp"  
 android:gravity="top"  
 android:text="x"  
 android:textAppearance="@style/x" />  
  
  
 <Button  
 android:id="@+id/check\_add"  
 android:layout\_width="116dp"  
 android:layout\_height="53dp"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentTop="true"  
 android:layout\_marginLeft="120dp"  
 android:layout\_marginTop="570dp"  
 android:background="@drawable/input\_border2"  
 android:text="확인"  
  
 />  
  
 <EditText  
 android:id="@+id/className"  
 android:layout\_width="270dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="40dp"  
 android:layout\_marginTop="311dp"  
 android:paddingStart="10dp"  
 android:paddingEnd="10dp"  
 android:textColorHint="#AAAAAA"  
 android:textColor="#FFFFFF"  
 android:autofillHints=""  
 android:background="@drawable/search\_box"  
 android:ems="10"  
  
 android:inputType="text"  
 android:minHeight="40dp" />  
  
</RelativeLayout>

Implement the interface by inserting a TextView and an OK button to receive the classroom name and array.

[AddActivity]

package com.example.success;  
  
import androidx.activity.result.ActivityResultLauncher;  
import androidx.activity.result.contract.ActivityResultContracts;  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.core.app.ActivityCompat;  
  
import android.content.Intent;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.Button;  
  
public class MainActivity extends AppCompatActivity {  
  
 Button newClass;  
 String[] numbers;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
  
 newClass = findViewById(R.id.*newClass6*);  
 numbers = new String[]{};  
  
  
 findViewById(R.id.*btn\_add*).setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 Intent intent = new Intent(MainActivity.this,AddActivity.class);  
 getAddActivityResult.launch(intent);  
 }  
 });  
  
  
  
 }  
  
 private final ActivityResultLauncher<Intent> getAddActivityResult = registerForActivityResult(  
 new ActivityResultContracts.StartActivityForResult(),  
 result -> {  
 if(result.getResultCode() == *RESULT\_OK*){  
  
 newClass.setText(result.getData().getStringExtra("className"));  
 newClass.setVisibility(newClass.*VISIBLE*);  
 numbers = result.getData().getStringArrayExtra("numbers");  
  
 }  
 }  
 );  
  
 public void gotoselect(View view){  
 Intent intent2 = new Intent(MainActivity.this,SelectActivity.class);  
 //intent2.putExtra("numbers",numbers);  
 startActivity(intent2);  
 finish();  
 }  
  
  
}

AddActivity transfers the classroom name and array entered by the user to the main screen.

If you select a newly shown classroom on the main screen, it goes to selectAcitivity to select a desk.

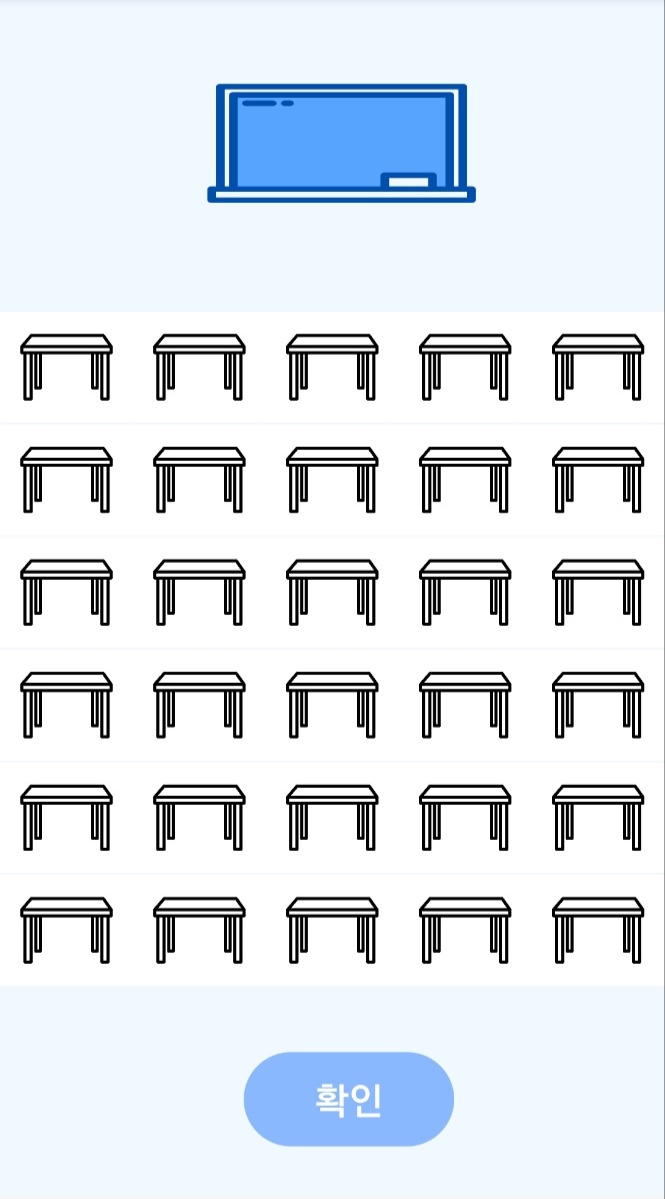
[activity\_select.xml]

<?xml version="1.0" encoding="utf-8"?>  
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:background="#F0F9FF">  
  
  
 <Button  
 android:id="@+id/imageView0"  
 android:layout\_width="150dp"  
 android:layout\_height="96dp"  
 android:layout\_marginTop="30dp"  
 android:layout\_marginLeft="110dp"  
  
 android:background="@drawable/chalkboard"  
 android:onClick="gotoNew"/>  
  
 <androidx.recyclerview.widget.RecyclerView  
 android:id="@+id/recycler\_view"  
 android:layout\_width="match\_parent"  
 android:layout\_height="390dp"  
 android:layout\_marginTop="170dp"/>  
  
 <Button  
 android:id="@+id/gotoresult"  
 android:layout\_width="116dp"  
 android:layout\_height="53dp"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentTop="true"  
 android:layout\_marginLeft="131dp"  
 android:layout\_marginTop="570dp"  
 android:background="@drawable/input\_border2"  
 android:text="확인"  
  
 />  
  
</RelativeLayout>

[table\_list.xml]

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginBottom="1dp"  
 android:background="#FFF"  
 android:clickable="true"  
 android:orientation="vertical">  
  
 <ImageView  
 android:id="@+id/imgview"  
 android:layout\_width="match\_parent"  
 android:layout\_height="60dp"  
 android:padding="5dp"  
 android:background="#00ff0000"  
 />  
  
  
  
</LinearLayout>

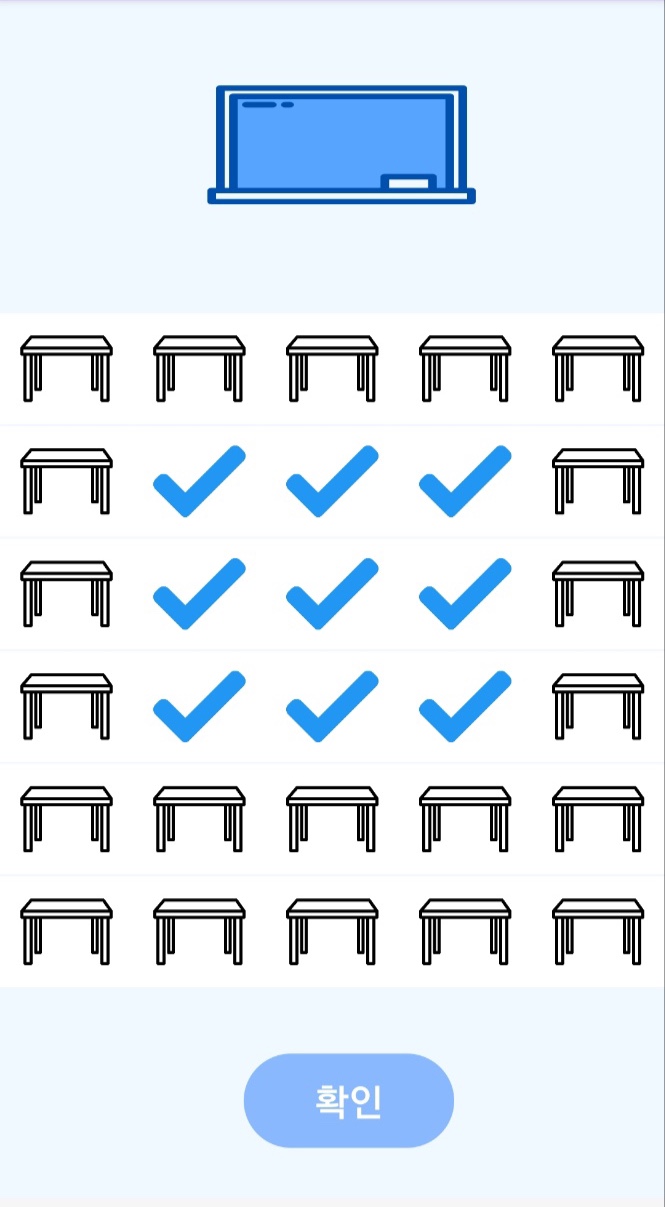
In Activity\_select.xml, one tablelist is reused.



[RecyclerViewAdapter.java]

package com.example.success;  
  
import android.graphics.Color;  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.widget.ImageView;  
import android.widget.TextView;  
  
import androidx.recyclerview.widget.RecyclerView;  
  
import java.util.List;  
  
  
  
public class RecyclerViewAdapter extends RecyclerView.Adapter<RecyclerViewAdapter.MyViewHolder> {  
  
 public List<Model> mModelList;  
  
 //추가  
 int position;  
  
 public RecyclerViewAdapter(List<Model> modelList) {  
 mModelList = modelList;  
 }  
  
  
 @Override  
 public MyViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {  
 View view = LayoutInflater.*from*(parent.getContext()).inflate(R.layout.*table\_list*, parent, false);  
 ItemClickListener itemClickListener = new ItemClickListener() {  
 @Override  
 public void onItemClick(int position) {  
  
 }  
 };  
 return new MyViewHolder(view,itemClickListener);  
 }  
  
 @Override  
 public void onBindViewHolder(final MyViewHolder holder, int position) {  
 final Model model = mModelList.get(position);  
  
 holder.imageView.setImageResource(R.drawable.*table*);  
  
 //추가 , 특정뷰에 태그를 달아준다  
 holder.imageView.setTag(position);  
  
 holder.imageView.setImageResource(model.isSelected() ? R.drawable.*check*: R.drawable.*table*);  
 holder.imageView.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 model.setSelected(!model.isSelected());  
 //holder.imageView.setImageResource(R.drawable.check);  
 //view.setBackgroundColor에서 바꿈 확인할것 -> 잘 작동  
 holder.imageView.setImageResource(model.isSelected() ? R.drawable.*check*: R.drawable.*table*);  
 }  
 });  
 }  
  
 @Override  
 public int getItemCount() {  
 return mModelList == null ? 0 : mModelList.size();  
 }  
  
 public class MyViewHolder extends RecyclerView.ViewHolder implements View.OnClickListener {  
  
 View view;  
 ImageView imageView;  
 ItemClickListener itemClickListener;  
  
 public MyViewHolder(View itemView,ItemClickListener itemClickListener) {  
 super(itemView);  
 view = itemView;  
 imageView = itemView.findViewById(R.id.*imgview*);  
  
 this.itemClickListener = itemClickListener;  
 imageView.setOnClickListener(this);  
 }  
  
 @Override  
 public void onClick(View v){  
 itemClickListener.onItemClick(getAbsoluteAdapterPosition());  
 }  
 }  
  
 public interface ItemClickListener{  
 void onItemClick(int position);  
 }  
  
  
}

Implement a function that returns a checkmark when a desk is selected.



[SelectActivity]

package com.example.success;  
  
import androidx.annotation.NonNull;  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.recyclerview.widget.GridLayoutManager;  
import androidx.recyclerview.widget.LinearLayoutManager;  
import androidx.recyclerview.widget.RecyclerView;  
  
import android.content.Intent;  
import android.graphics.Color;  
import android.graphics.ColorSpace;  
import android.os.Bundle;  
import android.util.Log;  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.widget.Button;  
import android.widget.ImageButton;  
import android.widget.ImageView;  
import android.widget.TextView;  
  
import java.util.ArrayList;  
import java.util.LinkedList;  
import java.util.List;  
import java.util.Queue;  
import java.util.Scanner;  
  
//전략1 알고리즘  
  
  
public class SelectActivity extends AppCompatActivity {  
  
  
 List<Model> mModelList;  
 List<Model> newList;  
 RecyclerView mRecyclerView;  
 RecyclerView.Adapter mAdapter;  
  
 Button gotoresult;  
 RecyclerViewAdapter.ItemClickListener itemClickListener;  
 int pos;  
 String str\_pos;  
 int ans;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_select*);  
  
 mRecyclerView = (RecyclerView) findViewById(R.id.*recycler\_view*);  
 gotoresult = findViewById(R.id.*gotoresult*);  
 mAdapter = new RecyclerViewAdapter(getListData());  
  
 GridLayoutManager manager = new GridLayoutManager(SelectActivity.this,5);  
 mRecyclerView.setHasFixedSize(true);  
 mRecyclerView.setLayoutManager(manager);  
 mRecyclerView.setAdapter(mAdapter);  
  
 newList = new ArrayList<>();  
  
 //선택된 책상 position을 받음  
 itemClickListener = new RecyclerViewAdapter.ItemClickListener() {  
 @Override  
 public void onItemClick(int position) {  
 pos = (int) mRecyclerView.getTag();  
 Log.*e*("테이블의 pos값 ",String.*valueOf*(pos));  
 str\_pos = String.*valueOf*(pos);  
  
 }  
 };  
  
 //ans => check(배열)  
 if(ans == 0){  
 //거리두기가 지켜지고 있지 않음  
 gotoresult.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 Intent intent = new Intent(SelectActivity.this,WarningActivity.class);  
 startActivity(intent);  
 finish();  
 }  
 });  
 }else if(ans == 1){  
 //거리두기가 지켜지고 있음  
 gotoresult.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 Intent intent = new Intent(SelectActivity.this,NewActivity.class);  
 startActivity(intent);  
 finish();  
 }  
 });  
 }  
  
  
 }  
  
 private List<Model> getListData(){  
 //책상을 담을 arraylist생성  
 mModelList = new ArrayList<>();  
 //입력받은 개수만큼 책상추가  
 for(int i=1;i<=30;i++){  
 mModelList.add(new Model(i));  
 }  
 return mModelList;  
  
 }  
  
  
  
 final static int *row* = 3;  
 final static int *col* = 6;  
  
 public static void main(String[] args) {  
 // *TODO Auto-generated method stub* Scanner kb = new Scanner(System.*in*);  
  
  
 String[] arr = new String[*row*];  
  
 // input the table information  
 // 유저로부터 테이블 상태 입력받기  
 for (int i = 0; i < *row*; i++) {  
 arr[i] = kb.nextLine();  
 }  
  
 System.*out*.println();  
  
 int ans;  
  
 // solution function  
 ans = *solution*(arr);  
  
 System.*out*.print(ans + " ");  
  
 }  
  
  
  
 public static int solution(String[] places) {  
 int answer;  
  
 String[] p = places;  
  
 // isOk 변수  
 // 거리두기가 지켜지고 있는지 확인하기 위한 변수  
 boolean isOk = true;  
  
 // 모든 자리에서 bfs 실행  
 // 사람이 앉아있는 자리 중, 한 자리라도 거리두기가 지켜지지 않으면 false 반환  
 for (int r = 0; r < *row* && isOk; r++) {  
 for (int c = 0; c < *col* && isOk; c++) {  
 if (p[r].charAt(c) == 'P') {  
 if (!*bfs*(r, c, p))  
 isOk = false;  
 }  
 }  
 }  
  
 // 거리두기가 지켜지면 1, 지켜지지 않으면 0 반환  
 answer = isOk ? 1 : 0;  
  
 return answer;  
 }  
  
 // 각 자리에서 거리두기를 확인하기 위한 bfs 함수  
 private static boolean bfs(int r, int c, String[] p) {  
 int dr[] = { -1, 1, 0, 0 };  
 int dc[] = { 0, 0, -1, 1 };  
  
 // 거리두기 비교하기 위한 queue  
 Queue<int[]> queue = new LinkedList<int[]>();  
 queue.offer(new int[] { r, c });  
  
 while (!queue.isEmpty()) {  
 int[] position = queue.poll();  
  
 // i가 0일 경우 왼쪽으로 1칸  
 // i가 1일 경우 오른쪽으로 1칸  
 // i가 2일 경우 위로 1칸  
 // i가 3일 경우 아래로 1칸  
 // for loop를 통해 맨하튼거리 = 1인 자리 비교  
 for (int i = 0; i < 4; i++) {  
 int nr = position[0] + dr[i];  
 int nc = position[1] + dc[i];  
  
 // 가장자리인 경우 빈자리, 거리두기 확인할 필요 없음  
 // 처음 시작점 P는 탐색 제외 (nr == r, nc == c)인 부분  
 if (nr < 0 || nc < 0 || nr >= *row* || nc >= *col* || (nr == r && nc == c))  
 continue;  
  
 // 각 자리로부터 맨하튼 거리 d  
 int d = Math.*abs*(nr - r) + Math.*abs*(nc - c);  
  
 // 파티션이 있는 경우 : 탐색 진행 X 더 이상 찾을 필요 없음  
 // 맨하튼거리가 2 이하이면서 다른 사람이 있는 경우 (P), false 반환  
 if (p[nr].charAt(nc) == 'P' && d <= 2)  
 return false;  
 // 맨하튼 거리가 2 미만이면서 빈자리인 경우 (O) 다음 탐색  
 // 다음 탐색을 할 경우 맨하튼 거리가 1 증가하므로 d < 2인 칸만 탐색  
 else if (p[nr].charAt(nc) == 'O' && d < 2)  
 queue.offer(new int[] { nr, nc });  
 }  
 }  
  
 return true;  
 }  
}

Implement strategy 1 algorithm that checks whether the entered seats keep distance. If the distance is maintained, it goes to NewActivity to display a screen, and to WarningActivity to display a warning window if the distance is not respected.

[activity\_warning.xml]

<?xml version="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 android:background="#F0F9FF">  
  
 <ImageView  
 android:id="@+id/imageView0"  
 android:layout\_width="179dp"  
 android:layout\_height="175dp"  
 android:layout\_marginTop="190dp"  
 android:src="@drawable/warning"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_chainStyle="spread"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent" />  
  
 <TextView  
 android:id="@+id/some\_id"  
 android:layout\_width="221dp"  
 android:layout\_height="58dp"  
 android:layout\_marginTop="408dp"  
 android:gravity="center"  
 android:text="거리두기가 지켜지지 않았습니다."  
 android:textColor="#566C81"  
 android:textSize="16sp"  
 android:textStyle="bold"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_chainStyle="spread"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent" />  
  
 <Button  
 android:id="@+id/next"  
 android:layout\_width="321dp"  
 android:layout\_height="58dp"  
 android:layout\_marginLeft="15dp"  
 android:layout\_marginTop="566dp"  
 android:background="#89B8FF"  
 android:textColor="#000000"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_chainStyle="spread"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 android:text="돌아가기"/>  
  
</androidx.constraintlayout.widget.ConstraintLayout>

Displays a warning through ImageView so that the user can see the warning window, and implements an interface with a return button to move to SelectActivity to select a seat again..

[WarningActivity]

package com.example.success;  
  
import android.content.Intent;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.Button;  
  
import androidx.annotation.Nullable;  
import androidx.appcompat.app.AppCompatActivity;  
  
public class WarningActivity extends AppCompatActivity {  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_warning*);  
  
 Button imageButton = (Button) findViewById(R.id.*next*);  
 imageButton.setOnClickListener(new View.OnClickListener() {  
  
 @Override  
 public void onClick(View view) {  
 Intent intent = new Intent(getApplicationContext(), SelectActivity.class);  
 startActivity(intent);  
 }  
 });  
 }  
}

Return to SelectActivity when the return button is pressed.

[NewActivity & activity\_new.xml]

package com.example.success;  
  
import androidx.appcompat.app.AppCompatActivity;  
  
import android.content.Intent;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.Button;  
  
import java.util.LinkedList;  
import java.util.Queue;  
import java.util.Scanner;  
  
//전략2 알고리즘  
  
public class NewActivity extends AppCompatActivity {  
  
 Button gotoMain;  
 //배열선언  
 String[] arr;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_new*);  
  
 Intent intent = getIntent();  
 //선언된 배열에 intent로 받아온 배열 넣어줌  
  
 //종료버튼 누르면 main으로 돌아감  
 gotoMain = findViewById(R.id.*gotomain*);  
 gotoMain.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 Intent intent = new Intent(NewActivity.this,MainActivity.class);  
 startActivity(intent);  
 finish();  
 }  
 });  
  
  
 }  
  
 final static int *row* = 6;  
 final static int *col* = 5;  
  
 public static int[][] *ptarr* = new int[*row*][*col*];  
  
  
 public static void main(String[] args) {  
 // *TODO Auto-generated method stub* Scanner kb = new Scanner(System.*in*);  
  
  
 String[] arr = new String[*row*];  
  
 // 파티션이 필요한 자리를 저장하기 위한 string 배열  
 for (int i = 0; i < *row*; i++) {  
 for (int j = 0; j < *col*; j++) {  
 *ptarr*[i][j] = 0;  
 }  
 }  
  
 // input the table information  
 // 유저로부터 테이블 상태 입력받기  
 for (int i = 0; i < *row*; i++) {  
 arr[i] = kb.nextLine();  
 }  
  
 System.*out*.println();  
  
 int ans;  
  
 ans = *solution*(arr);  
  
 // 함수를 통해 얻은 결과 출력  
 // 파티션이 필요한 자리 = 1, 그 외의 자리는 0  
 for (int i = 0; i < *row*; i++) {  
 for (int j = 0; j < *col*; j++) {  
 System.*out*.print(*ptarr*[i][j] + " ");  
 }  
 System.*out*.println();  
 }  
  
 }  
  
  
  
 public static int solution(String[] places) {  
 int answer;  
  
 String[] p = places;  
  
 // isOk 변수  
 // 거리두기가 지켜지고 있는지 확인하기 위한 변수  
 boolean isOk = true;  
  
 // 모든 자리에서 bfs 실행  
 // 사람이 앉아있는 자리 중, 한 자리라도 거리두기가 지켜지지 않으면 false 반환  
 for (int r = 0; r < *row* && isOk; r++) {  
 for (int c = 0; c < *col* && isOk; c++) {  
 if (p[r].charAt(c) == 'P') {  
 if (!*bfs*(r, c, p))  
 isOk = false;  
 }  
 }  
 }  
  
 // 거리두기가 지켜지면 1, 지켜지지 않으면 0 반환  
 answer = isOk ? 1 : 0;  
  
 return answer;  
 }  
  
 // 각 자리에서 거리두기를 확인하기 위한 bfs 함수  
 private static boolean bfs(int r, int c, String[] p) {  
 int dr[] = { -1, 1, 0, 0 };  
 int dc[] = { 0, 0, -1, 1 };  
  
 // 거리두기 비교하기 위한 queue  
 Queue<int[]> queue = new LinkedList<int[]>();  
 queue.offer(new int[] { r, c });  
  
 while (!queue.isEmpty()) {  
 int[] position = queue.poll();  
  
 // i가 0일 경우 위로 1칸  
 // i가 1일 경우 아래로 1칸  
 // i가 2일 경우 왼쪽으로 1칸  
 // i가 3일 경우 오른쪽으로 1칸  
 // for loop를 통해 맨하튼거리 = 1인 자리 비교  
 for (int i = 0; i < 4; i++) {  
 int nr = position[0] + dr[i];  
 int nc = position[1] + dc[i];  
  
 // 가장자리인 경우 빈자리, 거리두기 확인할 필요 없음  
 // 처음 시작점 P는 탐색 제외 (nr == r, nc == c)인 부분  
 if (nr < 0 || nc < 0 || nr >= *row* || nc >= *col* || (nr == r && nc == c))  
 continue;  
  
 // 각 자리로부터 맨하튼 거리 d  
 int d = Math.*abs*(nr - r) + Math.*abs*(nc - c);  
  
 // 파티션이 있는 경우 : 탐색 진행 X 더 이상 찾을 필요 없음  
  
 // 맨하튼거리가 2 이하이면서 다른 사람이 있는 경우(P) 원래는 false 반환하고 함수 종료했지만  
 // 나머지 자리들에서 파티션이 필요한 위치를 저장하기 위해 반환하지 않음  
 if (p[nr].charAt(nc) == 'P' && d <= 2) {  
 // i = 0인 경우, 위로 한칸 이동한 것이므로 1칸 아래에 파티션 필요  
 if (i == 0) {  
 *ptarr*[nr + 1][nc] = 1;  
 }  
 // i = 1인 경우, 아래로 한칸 이동한 것이므로 1칸 위에 파티션 필요  
 else if (i == 1) {  
 *ptarr*[nr - 1][nc] = 1;  
 }  
 // i = 2인 경우, 왼쪽으로 한칸 이동한 것이므로 1칸 오른쪽에 파티션 필요  
 else if (i == 2) {  
 *ptarr*[nr][nc + 1] = 1;  
 }  
 // i = 3인 경우, 오른쪽으로 한칸 이동한 것이므로 1칸 왼쪽에 파티션 필요  
 else if (i == 3) {  
 *ptarr*[nr][nc - 1] = 1;  
 }  
 // return false;  
 }  
 // 맨하튼 거리가 2 미만이면서 빈자리인 경우 (O) 다음 탐색  
 // 다음 탐색을 할 경우 맨하튼 거리가 1 증가하므로 d < 2인 칸만 탐색  
 else if (p[nr].charAt(nc) == 'O' && d < 2)  
 queue.offer(new int[] { nr, nc });  
 }  
 }  
  
 return true;  
 }  
}

NewActivity finds a place for the screen through the strategy 2 algorithm implementation.