B & B Game Library

SUBMITTED TO :

Mrs. Anu Chauhan

SUBMITTED BY :

Bhav Beri

Xth ‘ B ’

ROLL NO. 4

CERTIFICATE

This is to certify that the project entitled "**B & B GAME LIBRARY**" is a record of work done by Bhav Beri, Class Xth 'B', Roll No. 4, in partial fulfillment of the requirements for the ICSE Computer Applications under the prestigious school "**St. Joseph's Convent School, Phagwara** " in the session 2018-19.

Internal Examiner

External Examiner

ACKNOWLEDGEMENT

I would like to express my special gratitude to my teacher, **Mrs. Anu Chauhan** who guided me in the completion of this project. Her cooperation, suggestions and guidance helped me in successful and timely completion of this project. I would also like to thank our principal who gave me the golden opportunity to do this wonderful project on the topic '**B & B Game Library** ', which also helped me in doing a lot of research and learning new concepts and applications.

My gratitude goes to my parents, my grandparents, my sister and my friends who helped me a lot in finalizing this project within the limited time frame.

Bhav Beri

Xth ' B '

INDEX

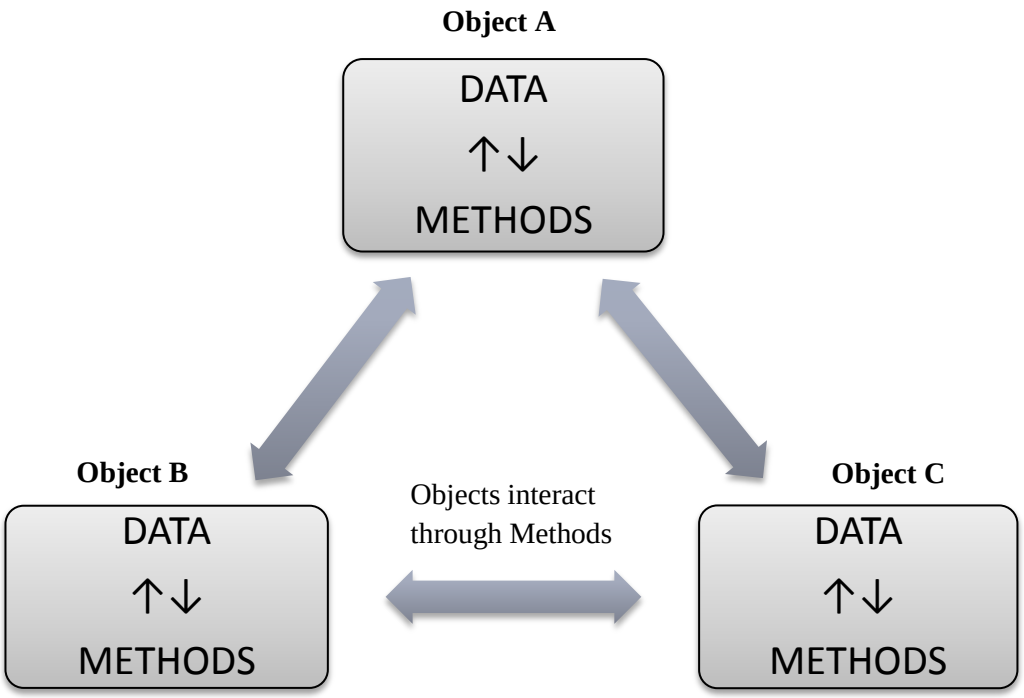
Serial Number	Content	Page Numbers
1	Introduction to OOP	1
2	Features of OOP	2 - 3
3	Introduction to Java	4
4	Features of Java	5 - 6
5	Introduction to BlueJ	7
6	Algorithm	8 - 10
7	Flowchart	11 - 59
8	Source Code	60 - 178
9	Output	179 - 191

Introduction to OOP

Over the years computer programming and computer languages have been continuously evolving based on the applications to be supported. We have different approaches like Procedural approach, Structured approach, and Modular approach. But the latest approach is the Object Oriented Approach (OOP).

OOP is a method where programming is adjusted and modeled on the basis of real life entities and processes. Object Oriented programming is a programming style which is associated with the concepts like class, object, Inheritance, Encapsulation, Abstraction, Polymorphism. OOP allows the programmer to create data variables or objects and processes. And these can be used and reused any number of times. It treats data as an important element and does not allow it to move freely in the program. It divides a program into building blocks called objects and then builds data and procedures/methods around these objects. The objects interact with each other through procedures/objects to perform a task. The method of one object can access the data of same object, which means that data can't be accessed by external methods.

Most popular programming languages like Java, C++, C#, Ruby, etc. follow an object oriented programming paradigm.



Features of OOP

1. OBJECTS

   An object is an identifiable entity with some characteristics and behavior of its own.
   An object is a software unit of related variables and methods. These are used to model real world objects we find in everyday life.
   They may represent a person, a place, a bank account, a table of data or any item that the program has to handle. They may also represent user-defined data such as vector, time and lists.
   When the program is executed, the object interacts by sending message to one another.
   Example: A dog is an object because it has states like color, name, breed, etc. as well as behaviors like wagging the tail, barking, eating, etc.
2. CLASSES
   A class is a group of related objects that share common properties, relationships and behaviours. It is a logical entity.
   A class is the blueprint from which individual objects or instances are created.
   A class is an object factory.
   A class is also known as user defined data type.
   Every class consists of
   - Data members (variables)
   - Member functions (methods)

   Example: A Class called Fruit has three characteristics colour, flavor and origin. The Objects – Mango, Apple and Orange have the same charateristics but each has its own specific values.

3. DATA ENCAPSULATION
   The binding or wrapping up of data and member functions into a single unit called class is known as encapsulation.
   It makes the data inaccessible to the outside world. They can be accessed only by its methods, which are wrapped within the class. Methods serve as an interface between the data of the object and the program.
   Encapsulation also provides for data hiding or information hiding, which is the process of protecting data from direct access by the program.
4. DATA ABSTRACTION
   Abstraction refers to the act of representing essential features without including the background details or explanations.
   It helps in the clear separation between the abstract properties and the details of its implementation.
   Example: When we make phone call, we don't know the internal processing but we only need to know that how to begin it and how to end it.

5. POLYMORPHISM
   The ability to take more than one form is called as polymorphism.
   In Java, we use method overloading and method overriding to achieve polymorphism.
   A function can exhibit different behaviours in different objects of the same class. This behaviour can be dependent on the types or number of arguments or data passed to the function.
   Example: A single function named Area() can be used to handle different types and number of arguments. Like:
   Area(<shape>) → Argument
   Area(square) → int side
   Area(rectangle) → int length, int breadth
   Area(circle) → int radius
6. INHERITANCE
   Inheritance is the process by which objects of one class acquire/inherits the properties of objects of another class.
   Inheritance provides the feature of reusability. This means that additional features can be added to an already existing class without modifying it. This is possible when a new class is derived from the existing one.
   The new class will have the combined features of both the classes.
   Each derived class should define only those features that are unique to it.
   It is used to achieve runtime polymorphism.
   Example: Let us consider a class Plant. Three classes: Herb, Shrub and Tree are all part of the class Plant and have few common features. Each derived class shares common characteristics (Have root, shoot and leaves, bear flowers, etc.) with the Base class and will have the combined features of both the classes.
7. MESSAGE PASSING
   Objects interact and communicate with each other using methods. In software terms, method calls are normally considered as messages. Information is passed as parameters along with the method call. A message (method call) is a request for execution of a method.
   An object asks another object to perform an action by sending it a message. The requesting (sending) object is referred to as sender and the receiving object is referred to as receiver.
   Example: A Book object asks the Student object for its Title by sending it a message. The Student object returns the Title back to the Book object.

Introduction to Java

Java is a powerful object oriented programming language and it is in many aspects similar to C++. Java originated at Sun Microsystems, Inc. in 1991. It was conceived by James Gosling, Patrick Naughton, Chris Warth, Ed Frank, and Mike Sheridan at Sun Microsystems, Inc.

It follows all the principles and features of OOP: Abstraction, Encapsulation, Inheritance, and Polymorphism. Java has many features giving it an edge over other programming languages.

It is intended to let application developers "write once, run anywhere" (WORA), meaning that compiled Java code can run on all platforms that support Java without the need for recompilation.

Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of computer architecture.

Uses of Java are:-

- JSP - In Java, JSP (Java Server Pages) is used to create dynamic web pages, such as in PHP and ASP.
- Applets - Applets are another type of Java programs that are implemented on Internet browsers and are always run as part of a web document.
- J2EE - Java 2 Enterprise Edition is a platform-independent environment that is a set of different protocols and APIs and is used by various organizations to transfer data between each other.
- JavaBeans - This is a set of reusable software components that can be easily used to create new and advanced applications.
- Mobile - In addition to the above technology, Java is widely used in mobile devices nowadays, many types of games and applications are being made in Java.

Features of Java

1. SIMPLE
   Java is very easy to learn, and its syntax is simple, clean and easy to understand.

   Java language is a simple programming language because:

   - Java syntax is based on C
   - Java has removed many complicated and rarely-used features, for example, explicit pointers, operator overloading, etc.
   - There is no need to remove unreferenced objects.

2. OBJECT ORIENTED LANGUAGE
   Java is an object-oriented programming language. Everything in Java is an object. Object-oriented means we organize our software as a combination of different types of objects that incorporates both data and behavior.
   Object-oriented programming (OOPs) is a methodology that simplifies software development and maintenance by providing some rules.
   It follows all the main principles of OOP.

3. PLATFORM INDEPENDENT and PORTABLE
   Java is platform independent because it is different from other languages like C, C++, etc. which are compiled into platform specific machines while Java is a write once, run anywhere language.
   Java code is compiled by the compiler and converted into bytecode. This bytecode is a platform-independent code because it can be run on multiple platforms, i.e., Write Once and Run Anywhere(WORA).
   Java is portable because it facilitates you to carry the Java bytecode to any platform. It doesn't require any implementation.
4. SECURE

   Java is best known for its security. With Java, we can develop virus-free systems. Java is secured because:

   - No explicit pointer
   - Java Programs run inside a virtual machine sandbox
   - Classloader**:** Classloader in Java is a part of the Java Runtime Environment(JRE) which is used to load Java classes into the Java Virtual Machine dynamically. It adds security by separating the package for the

classes of the local file system from those that are imported from network sources.

- Bytecode Verifier: It checks the code fragments for illegal code that can violate access right to objects.
- Security Manager: It determines what resources a class can access such as reading and writing to the local disk.

5. ROBUST

Robust simply means strong. Java is robust because:

- It uses strong memory management.
- There is a lack of pointers that avoids security problems.
- There is automatic garbage collection in java which runs on the Java Virtual Machine to get rid of objects which are not being used by a Java application anymore.
- There are exception handling and the type checking mechanism in Java. All these points make Java robust.

6. ARCHITECTURE NEUTRAL
   Java is architecture neutral because there are no implementation dependent features, for example, the size of primitive types is fixed.
7. DISTRIBUTED
   Java is distributed because it facilitates users to create distributed applications in Java. RMI and EJB are used for creating distributed applications. This feature of Java makes us able to access files by calling the methods from any machine on the internet.
8. AUTOMATIC GARBAGE COLLECTION
   It is a feature of Java with which it prevents corruption of memory.
9. LIGHT WEIGHT CODE
   Large Applications can be created without much coding.
10. MULTI-THREADED
    A thread is like a separate program, executing concurrently. We can write Java programs that deal with many tasks at once by defining multiple threads. The main advantage of multi-threading is that it doesn't occupy memory for each thread. It shares a common memory area. Threads are important for multi-media, Web applications, etc.
11. DYNAMIC
    All objects are created during run time. The linking of data and methods to the objects are also done during run time. This is also known as dynamic binding. Java is also used to create dynamic web content for web based applications.

Introduction to BlueJ

BlueJ is an Integrated Development Environment (IDE) for the Java programming language, developed mainly for educational purposes, but also suitable for small-scale software development. It runs with the help of JDK (Java Development Kit).

An Integrated Development Environment (IDE) in computer science is a software application that provides comprehensive facilities to computer programmers for software development.

BlueJ is based on the Blue system which is developed at the University of Sydney and Monash University, Australia. Originally, the BlueJ project was started at the Monash University. But now, it is developed and maintained by a joint research group at the Deakin University, Australia and the University of Kent, UK. It is supported by Sun Microsystems.

BlueJ was developed to support the learning and teaching of object-oriented programming, and its design differs from other development environments as a result. The main screen graphically shows the class structure of an application under development, and objects can be interactively created and tested. This interaction facility, combined with a clean, simple user interface, allows easy experimentation with objects under development. Object-oriented concepts (classes, objects, communication through method calls) are represented visually and in its interaction design in the interface.

Major Features of BlueJ are:

- Fully Integrated environment
- Graphical and textual editing
- Graphical class structure display
- Interactive object creation
- Interactive object calls
- Interactive testing
- Ideal for beginners
- Easy-to-use interface
- Built-in editor, compiler, debugger, virtual machine, etc.

Algorithm

Step 1 : Start

Step 2 : Firstly, a window appears which executes the name of the program (B & B Game Library) slowly – slowly.

Step 3 : Then, the image of all the games are appeared one by one on windows to show the user which games are present in the application.

Step 4 : Then, a window appears which warns the user that the program is in trial mode. User can run only limited times of game/s. User is also given option to buy the full application from the link given. If not, user can continue in trial mode.

Step 5 : Then, a menu appears giving option to user to play different games. The options available would be :-

- Space Piano Game
- Tic Tac Toe Game
- Tiny Army Game
- Exit Button

In addition to these, there would be a welcome.

Step 6 : According to the option selected by user, steps to be followed are given below :-

Option 1 : Space Piano Game

1) A window opens of the game Space Piano with some tiles bearing white colour and black colour in one of the tile randomly in each row and below in the window, 'Your scores' and 'Time left' are also displayed.
2) When the user clicks the first black tile, time is started, if the user doesn't clicks the next black tile in the provided time, the game would end, and if he clicks, the process goes on.
3) The time keeps reducing after some specific scores to increase the difficulty level with increase in scores.
4) When the user clicks wrong tile, or when user's time is over in clicking a tile, the game is terminated and score window is opened showing the user's scores and the highest scores of the application in his pc/computer/laptop.
5) A relevant message is also shown according to the user's scores as compared to the highest scores.
6) Then, after some time, the score window is terminated, and the menu reappears and the times the user can play another or same game is decreased by one.

Option 2 : Tic Tac Toe Game

1) A window opens of the game Tic Tac Toe with a grid of 3x3. First turn is of player with symbol X (or simply player X).
2) He can click in any of the empty box of the grid.
3) When he has clicked, the it is the turn of player with symbol O (or simply player O). He can also click in any of the empty box of the grid.
4) The process continues and goes on.
5) The player who makes the first a straight line of similar three symbols (XXX or YYY), that player wins. The line can be horizontal, vertical or slanting.
6) If no player can win, the match is drawn.
7) When the result of the match comes out, the game is terminated and a result window is opened showing the result of the match played just before.
8) Then, after some time, the score window is terminated, and the menu reappears and the times the user can play another or same game is decreased by one.

Option 3 : Tiny Army Game

1) A window opens of the game Tic Tac Toe with four path for enemies to come. Each path has its own gun by which the user can attack the enemies. Enemies come on the path one by one and, in starting, slowly – slowly.
2) The user can kill the enemy by shooting the gun which is present at that path end. One enemy is killed by one shot.
3) The process goes on and the speed of the enemies keeps increasing after some specific scores to increase the difficulty level with increase in scores.
4) When the enemy comes in the camp (or where the gun is placed), the game is terminated and score window is opened showing the user's scores and the highest scores of the application in his pc/computer/laptop.
5) A relevant message is also shown according to the user's scores as compared to the highest scores.
6) Then, after some time, the score window is terminated, and the menu reappears and the times the user can play another or same game is decreased by one.

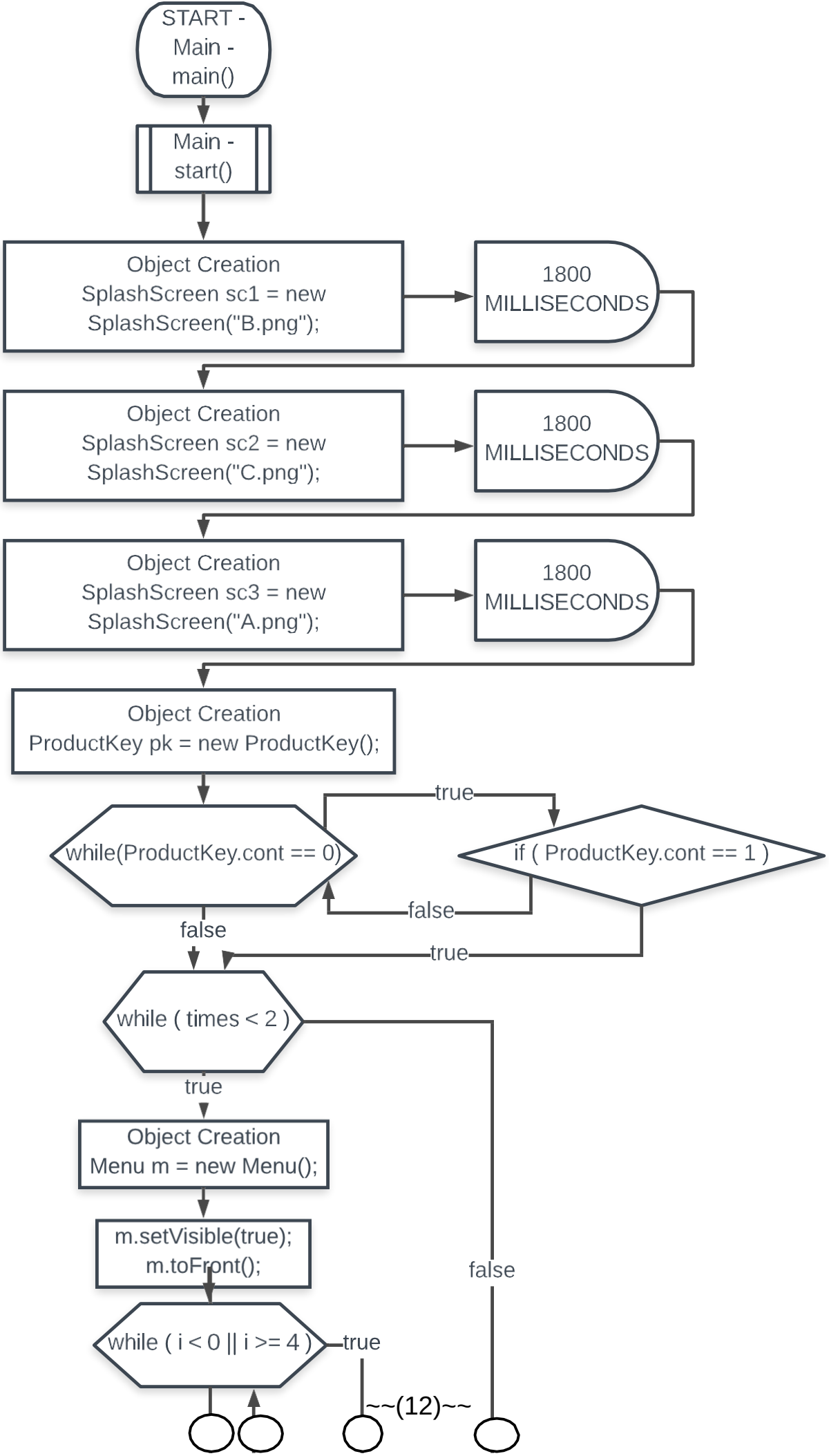
Option 4 : Exit Button or when the limited number of times the user can play the game have become zero.

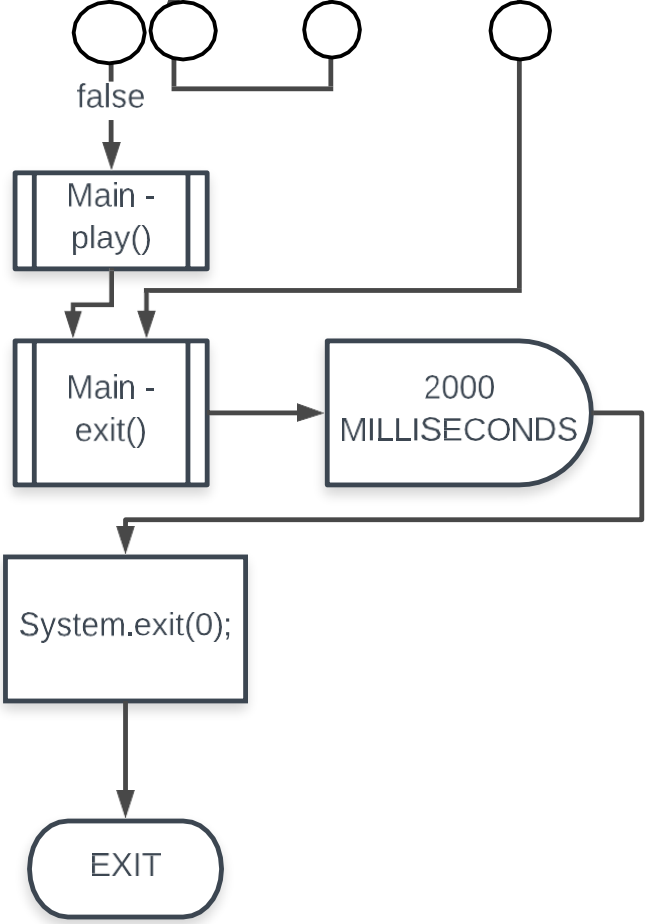
1) The menu window disappears.

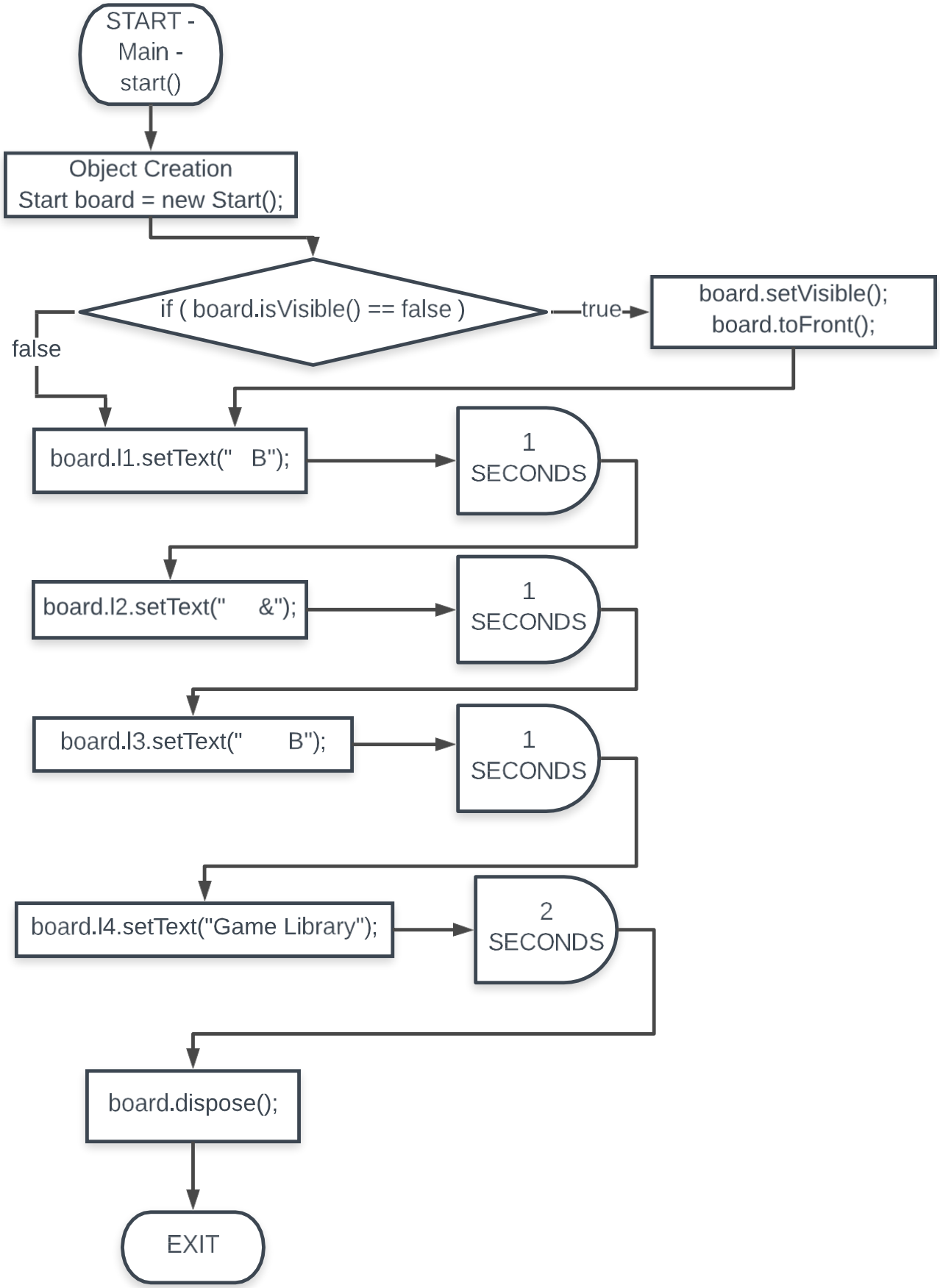
2) A new screen is opened, showing thanks for playing the game.
3) After some specific time, that window disappears.

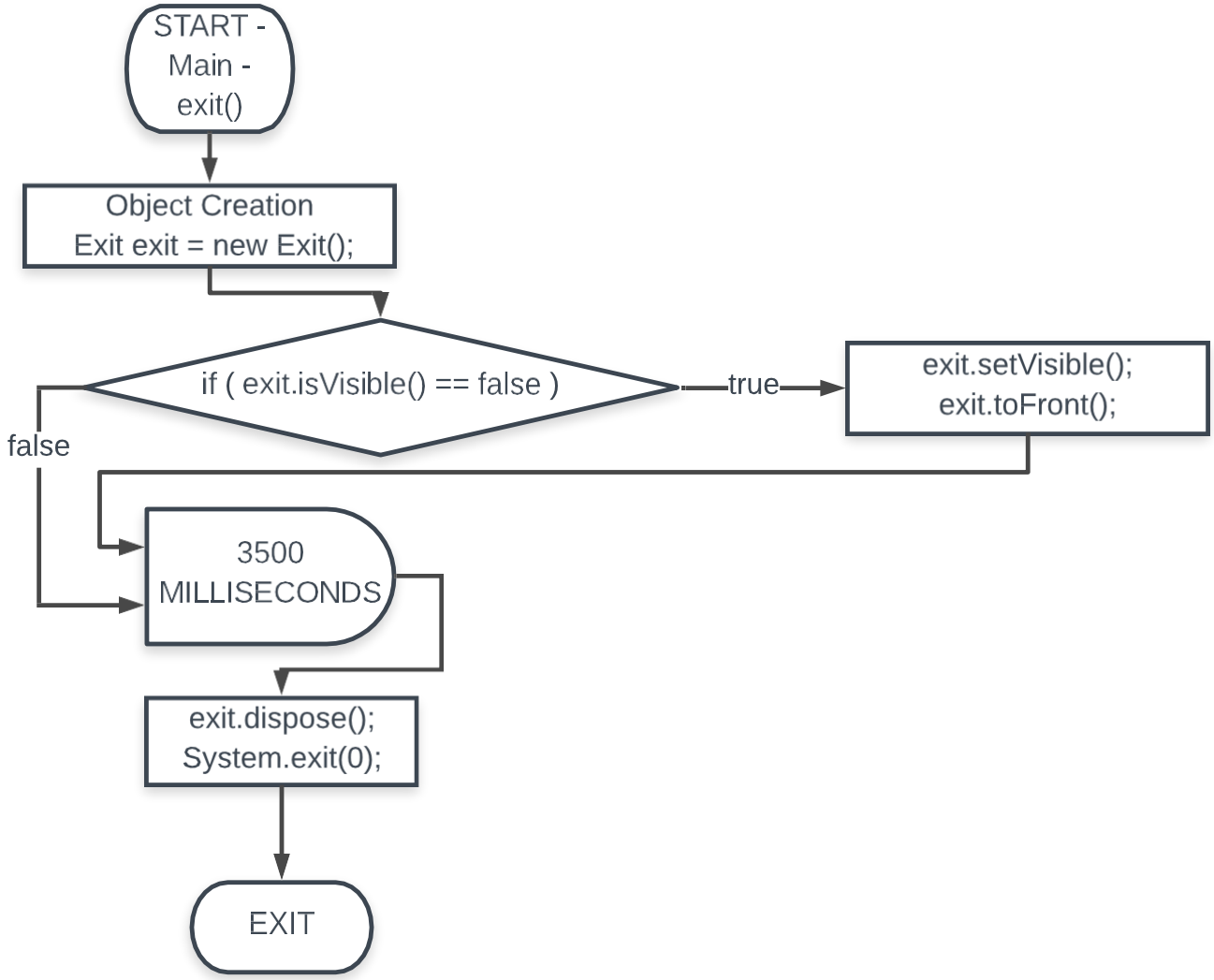
Step 7 : The Program is terminated.

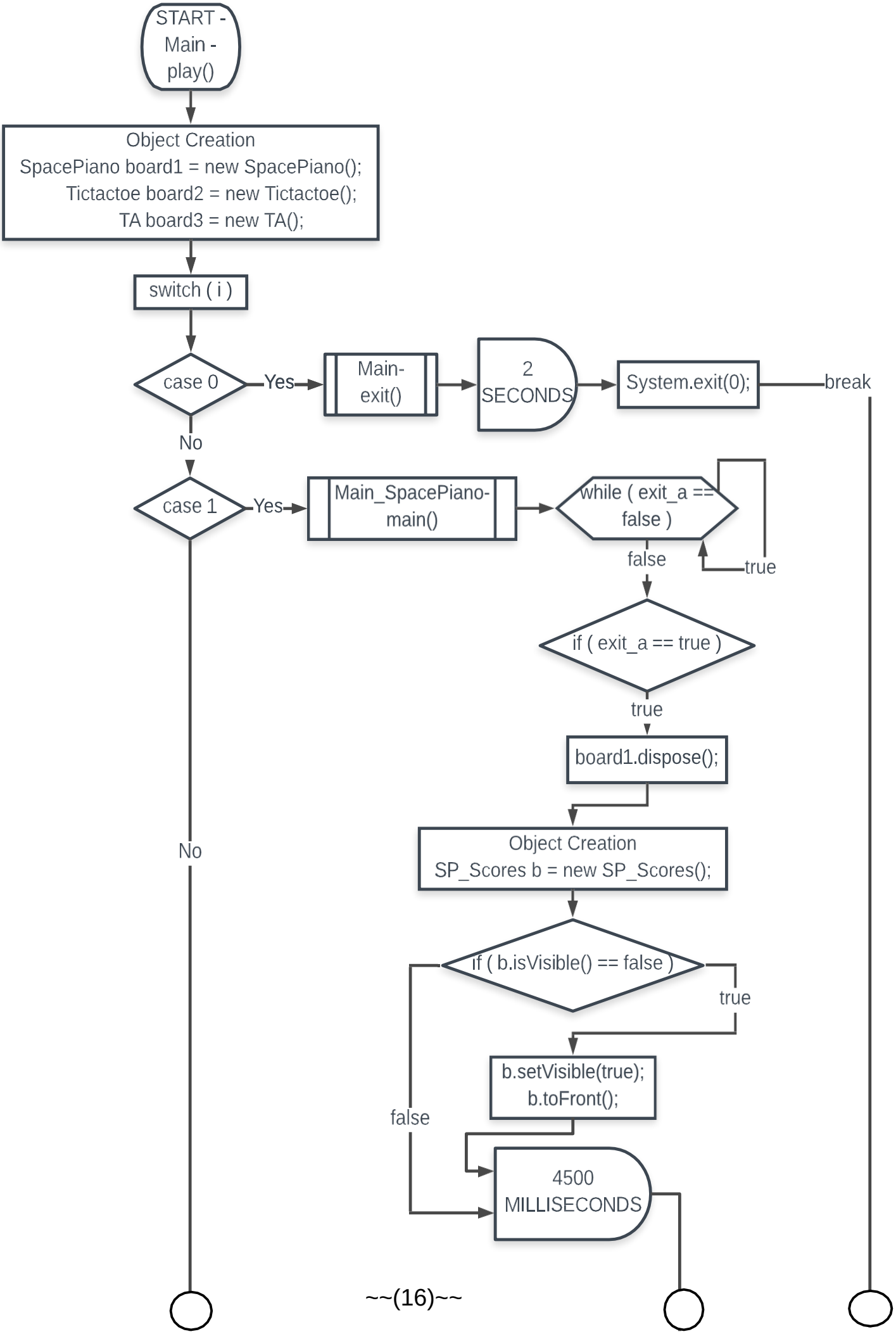
Flowchart

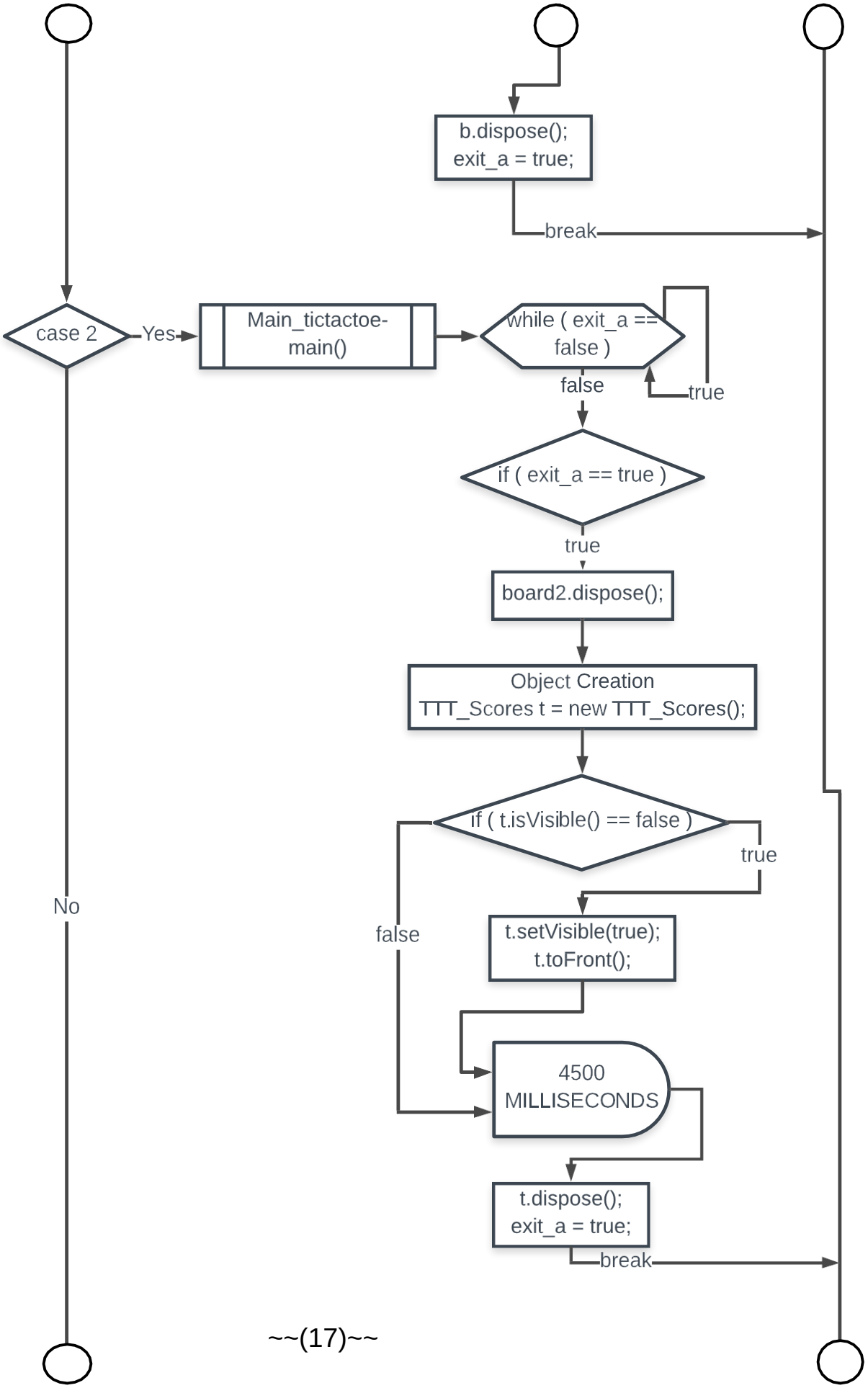

START - Main - main()
Main - start()
Object Creation SplashScreen sc1 = new SplashScreen("B.png");
1800 MILLISECONDS
Object Creation SplashScreen sc2 = new SplashScreen("C.png");
1800 MILLISECONDS
Object Creation SplashScreen sc3 = new SplashScreen("A.png");
1800 MILLISECONDS
Object Creation ProductKey pk = new ProductKey();
while(ProductKey.cont == 0)
true
if (ProductKey.cont == 1)
false
true
false
while (times < 2)
true
Object Creation Menu m = new Menu();
m.setVisible(true); m.toFront();
false
while (i < 0 || i >= 4)
true

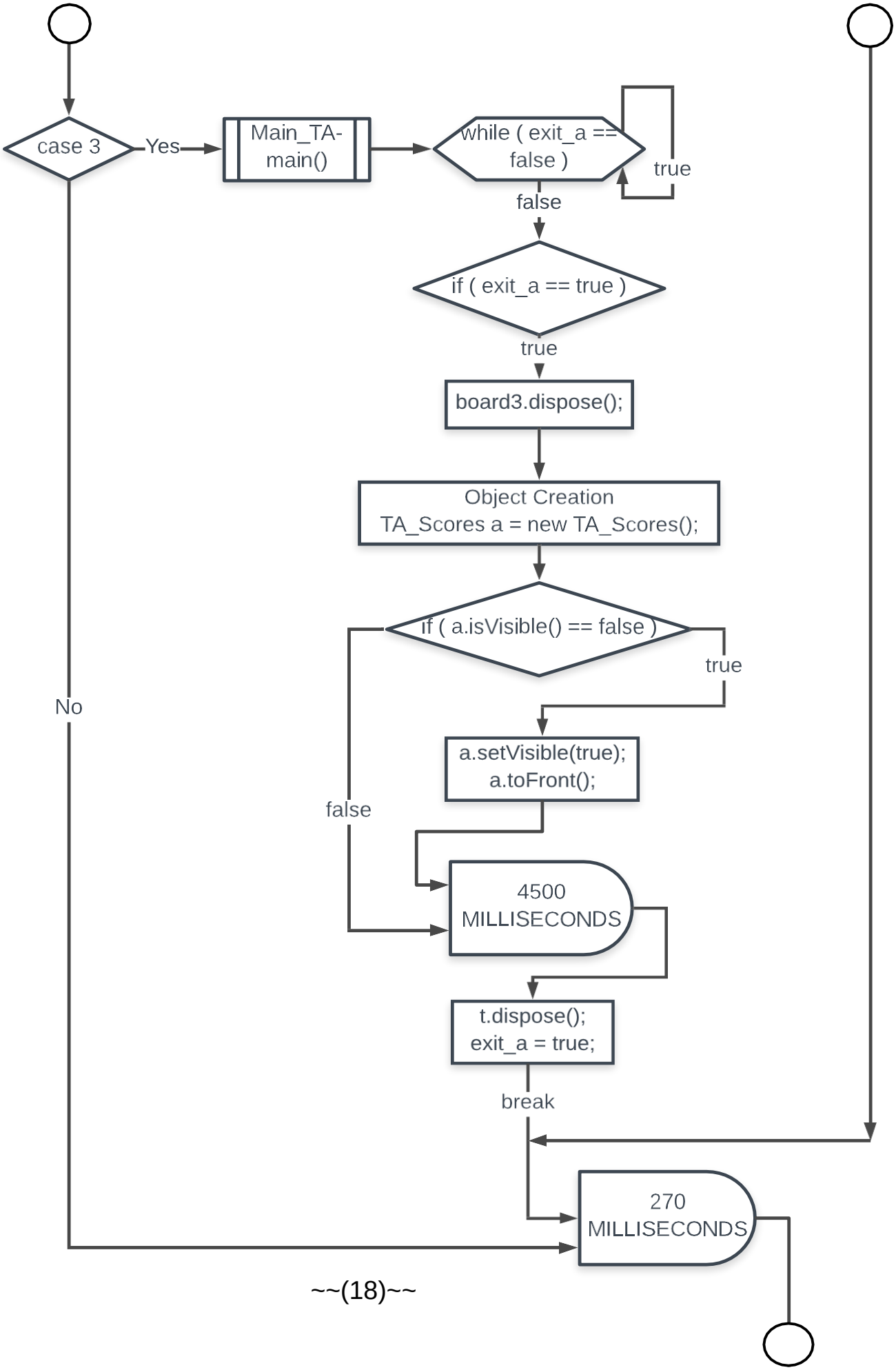

false
Main - play()
Main - exit()
2000 MILLISECONDS
System.exit(0);
EXIT

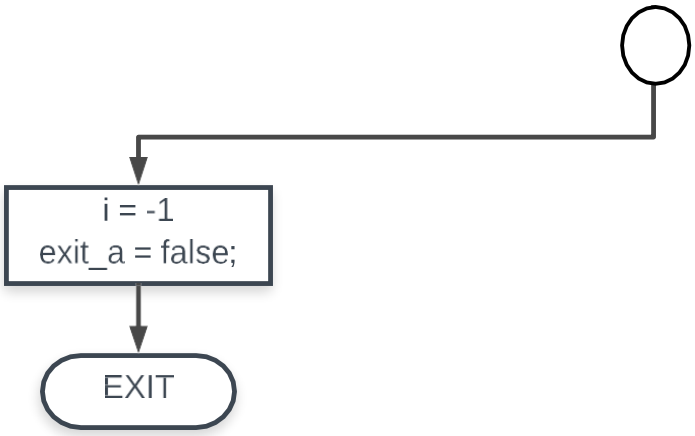

START -
Main -
start()
Object Creation
Start board = new Start();
if (board.isVisible() == false)
true
board.setVisible();
board.toFront();
false
board.l1.setText(" B");
1
SECONDS
board.l2.setText("   &");
1
SECONDS
board.l3.setText("   B");
1
SECONDS
board.l4.setText("Game Library");
2
SECONDS
board.dispose();
EXIT

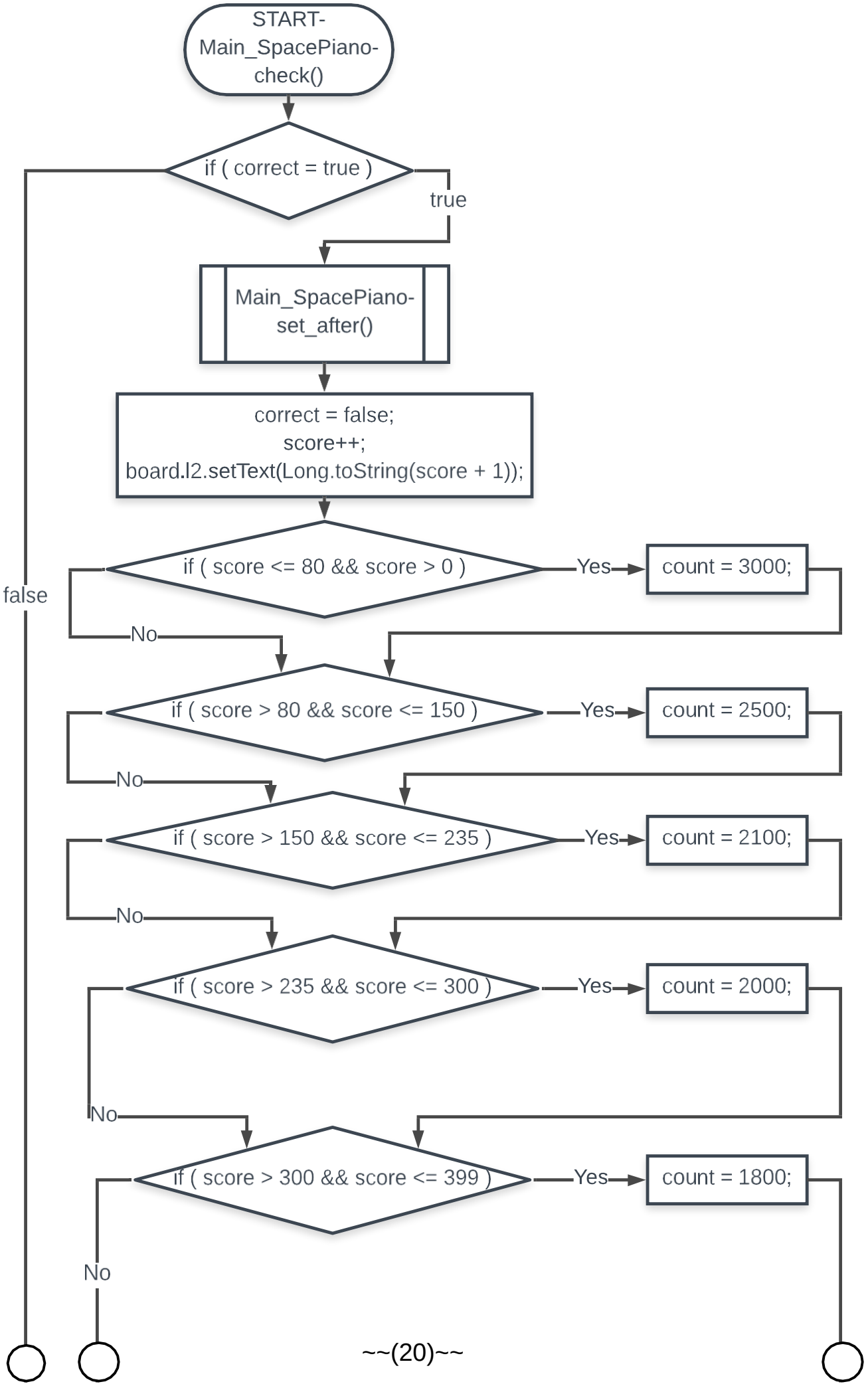

START -
Main -
exit()
Object Creation
Exit exit = new Exit();
if (exit.isVisible() == false)
true
exit.setVisible();
exit.toFront();
false
3500
MILLISECONDS
exit.dispose();
System.exit(0);
EXIT

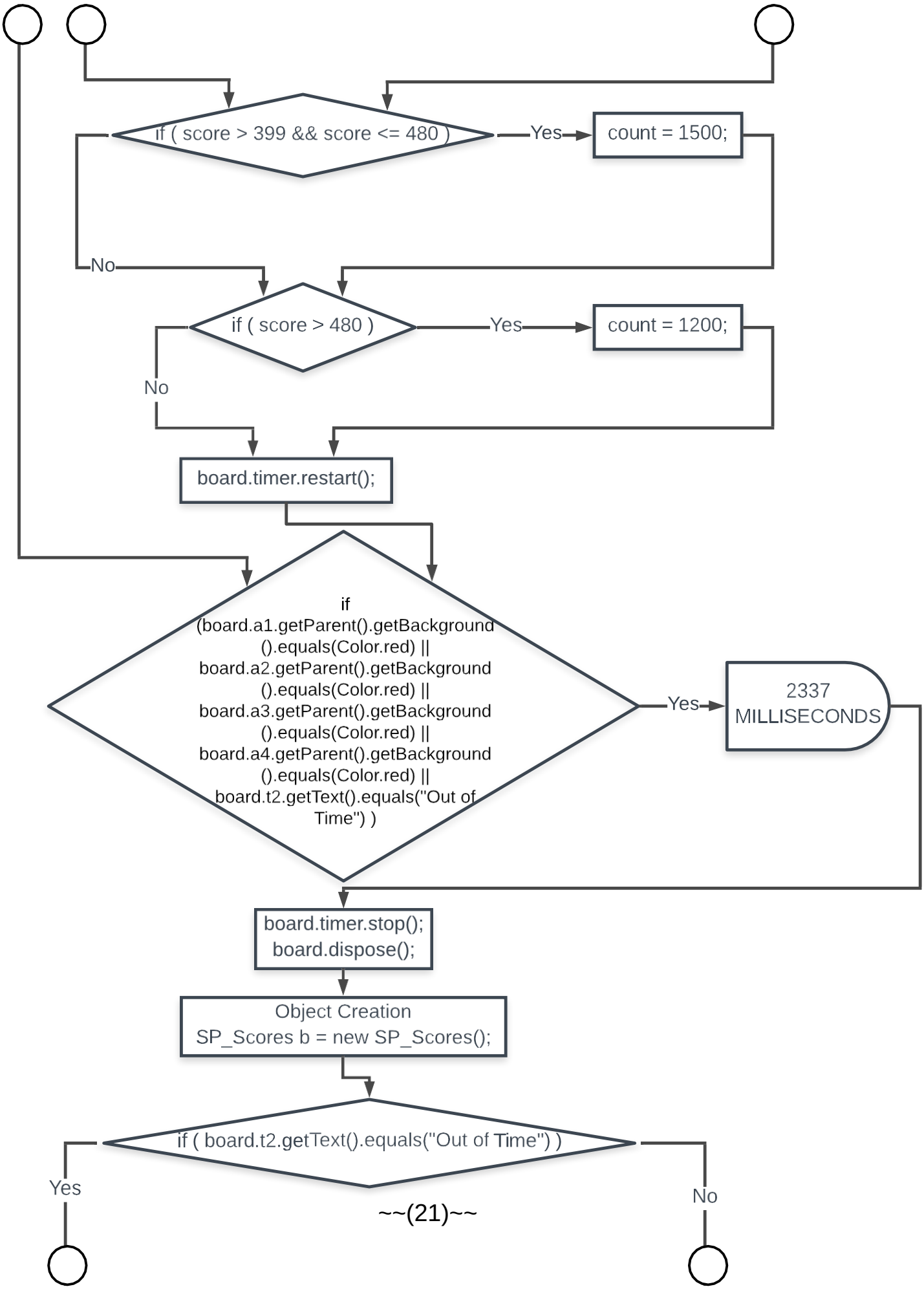

START - Main - play()
Object Creation
SpacePiano board1 = new SpacePiano();
Tictactoe board2 = new Tictactoe();
TA board3 = new TA();
switch (i)
case 0
Yes
Main- exit()
2 SECONDS
System.exit(0);
break
No
case 1
Yes
Main_SpacePiano- main()
while (exit_a == false)
false
true
if (exit_a == true)
true
board1.dispose();
No
Object Creation
SP_Scores b = new SP_Scores();
if (b.isVisible() == false)
true
b.setVisible(true);
b.toFront();
false
4500 MILLISECONDS

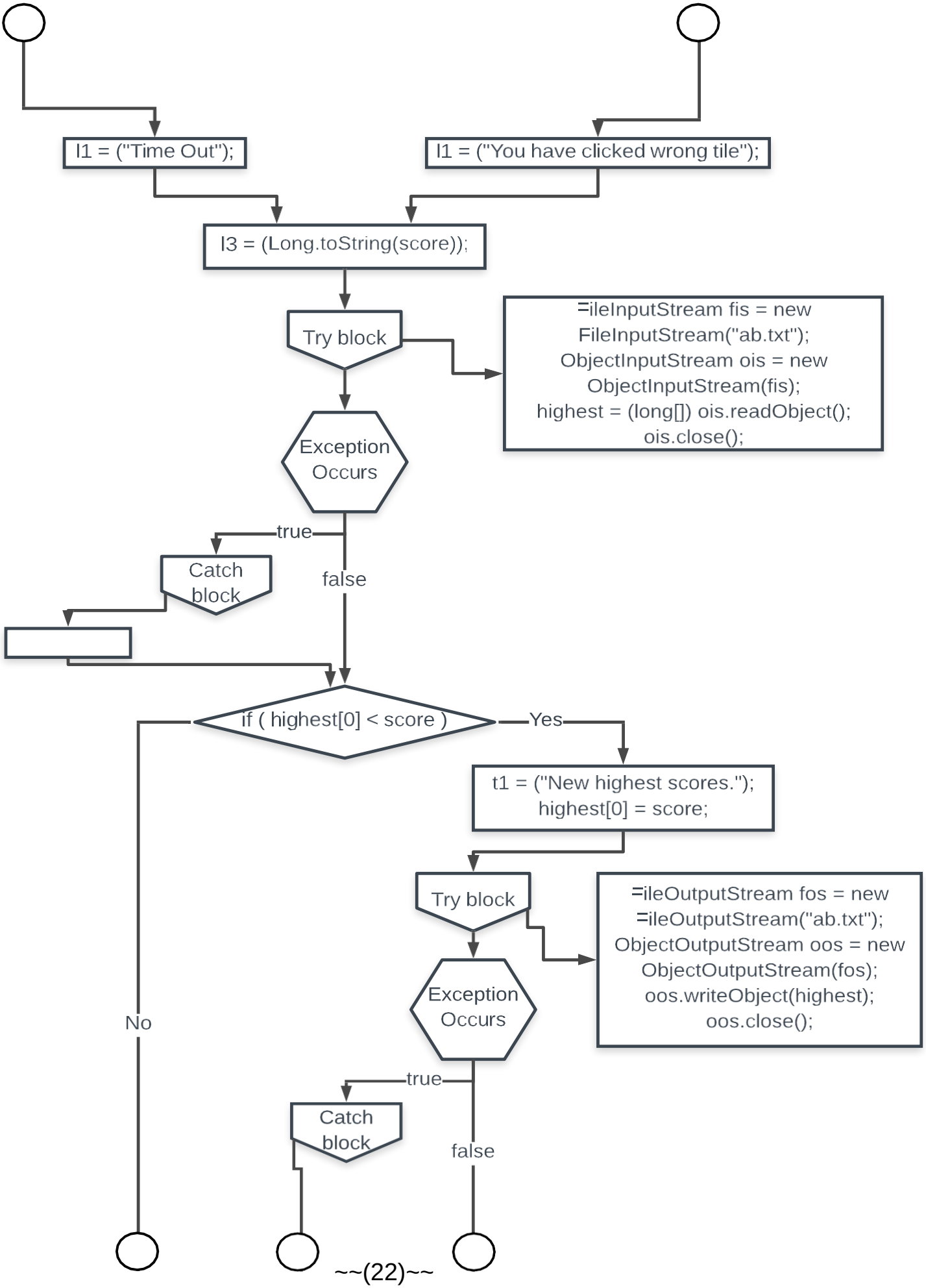

b.dispose();
exit_a = true;
break
case 2
Yes
Main_tictactoe-main()
while (exit_a == false)
false
true
if (exit_a == true)
true
board2.dispose();
Object Creation
TTT_Scores t = new TTT_Scores();
if (t.isVisible() == false)
true
false
t.setVisible(true);
t.toFront();
4500 MILLISECONDS
t.dispose();
exit_a = true;
break
No

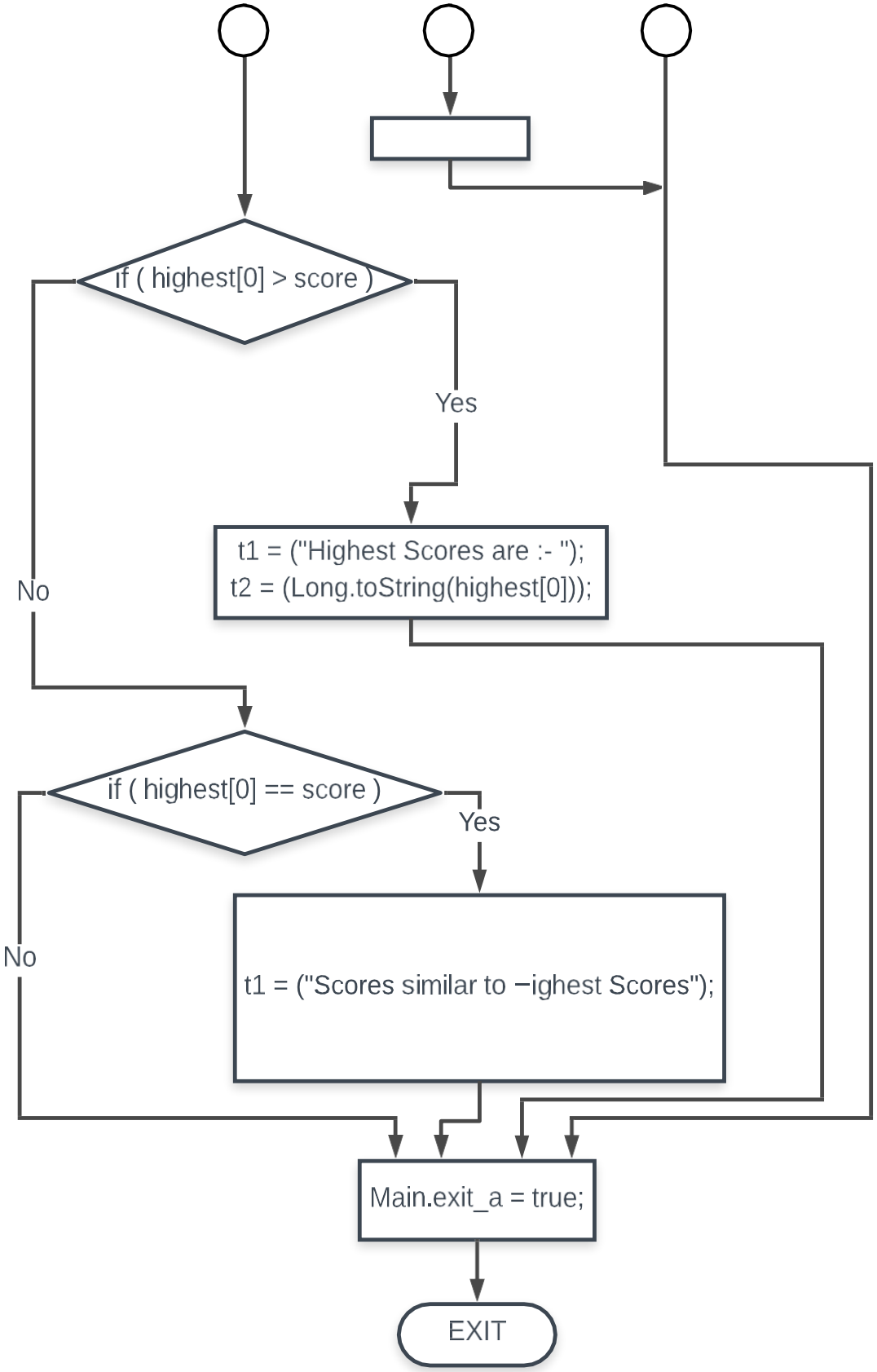

case 3
Yes
No
Main_TA-main()
while (exit_a == false)
true
false
if (exit_a == true)
true
board3.dispose();
Object Creation
TA_Scores a = new TA_Scores();
if (a.isVisible() == false)
true
false
a.setVisible(true);
a.toFront();
4500 MILLISECONDS
t.dispose();
exit_a = true;
break
270 MILLISECONDS

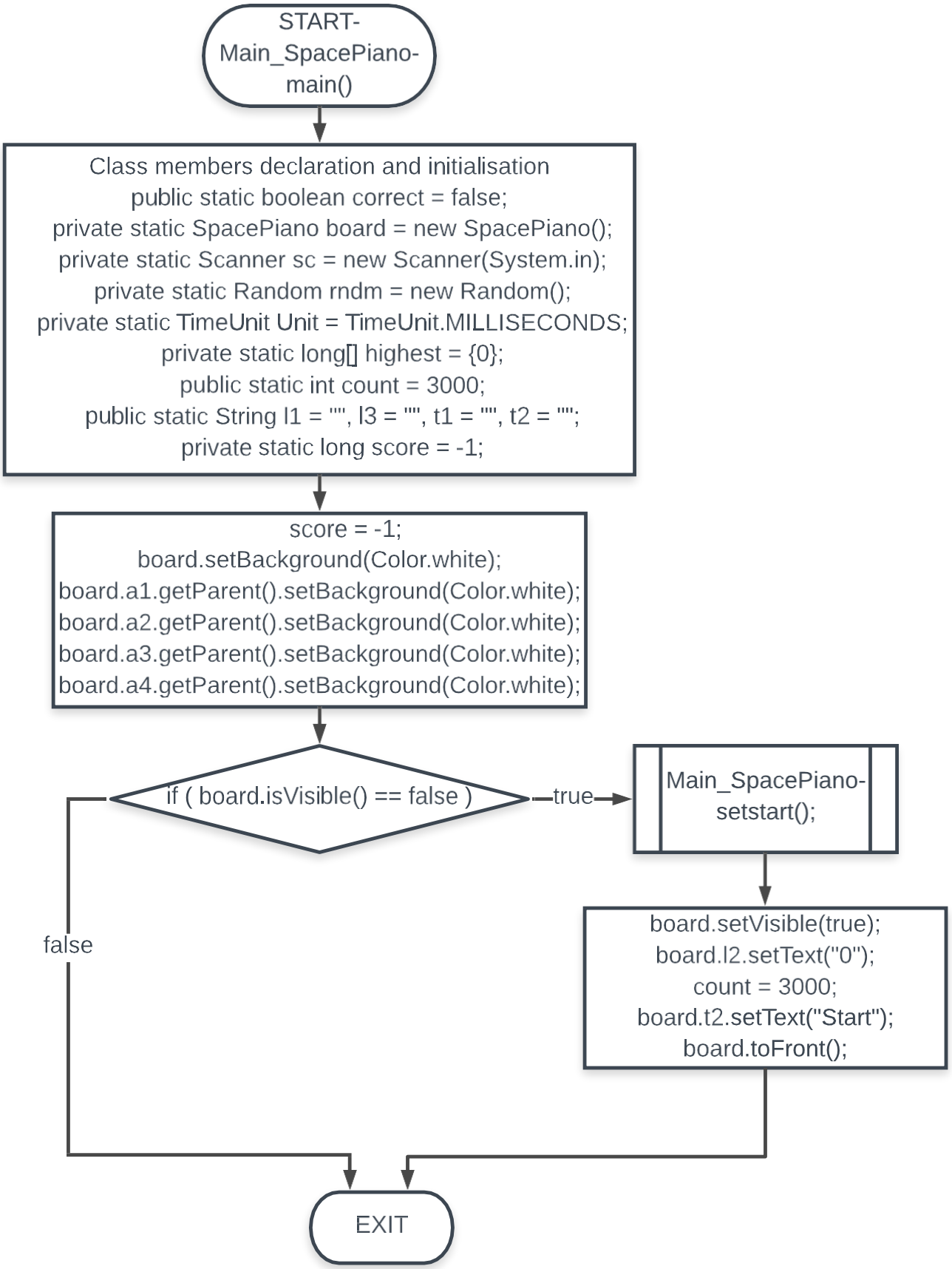

i = -1
exit_a = false;
EXIT

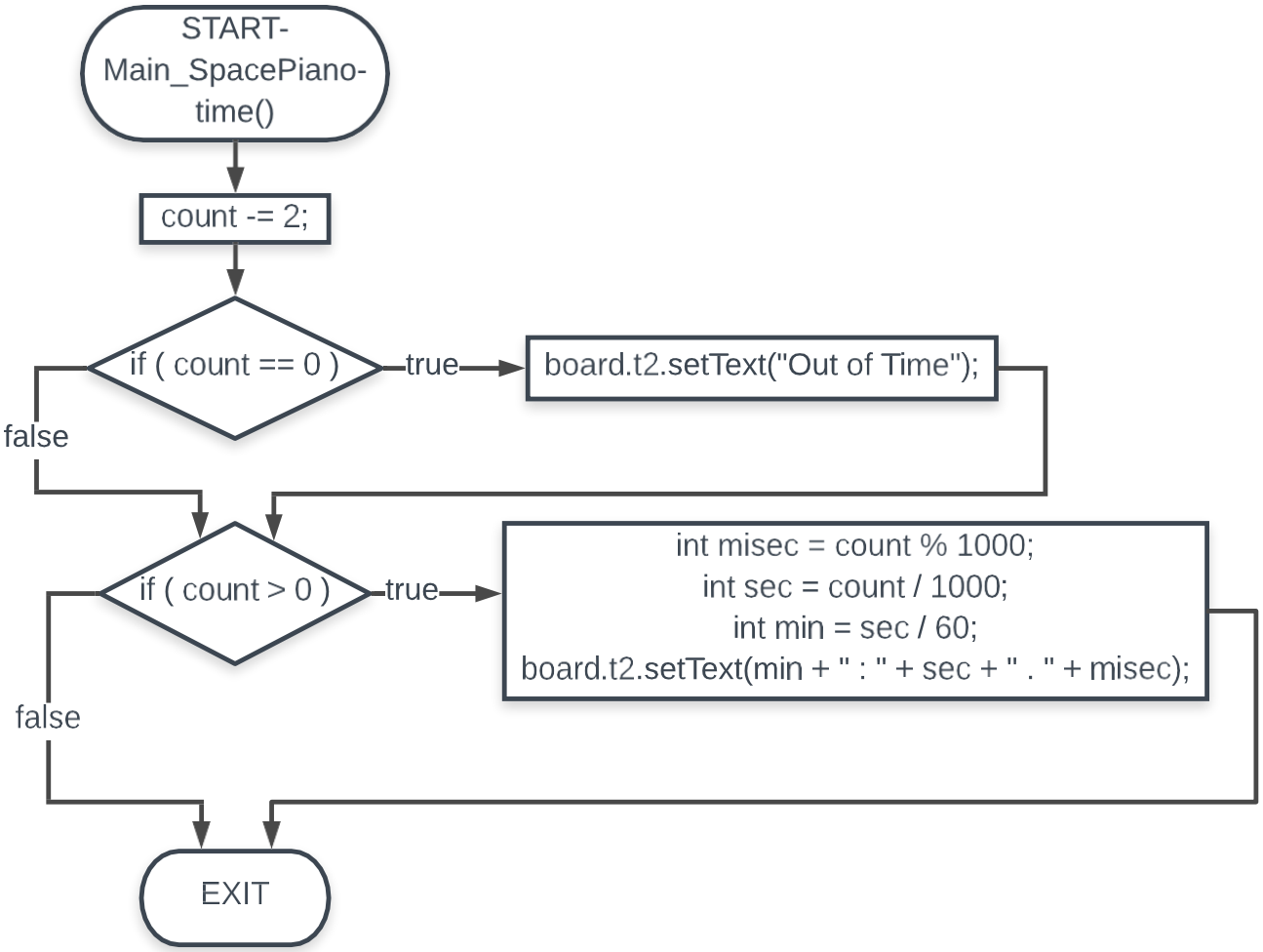

START-
Main_SpacePiano-
check()
if (correct = true)
true
false
Main_SpacePiano-
set_after()
correct = false;
score++;
board.l2.setText(Long.toString(score + 1));
if (score <= 80 && score > 0)
Yes
count = 3000;
No
if (score > 80 && score <= 150)
Yes
count = 2500;
No
if (score > 150 && score <= 235)
Yes
count = 2100;
No
if (score > 235 && score <= 300)
Yes
count = 2000;
No
if (score > 300 && score <= 399)
Yes
count = 1800;
No

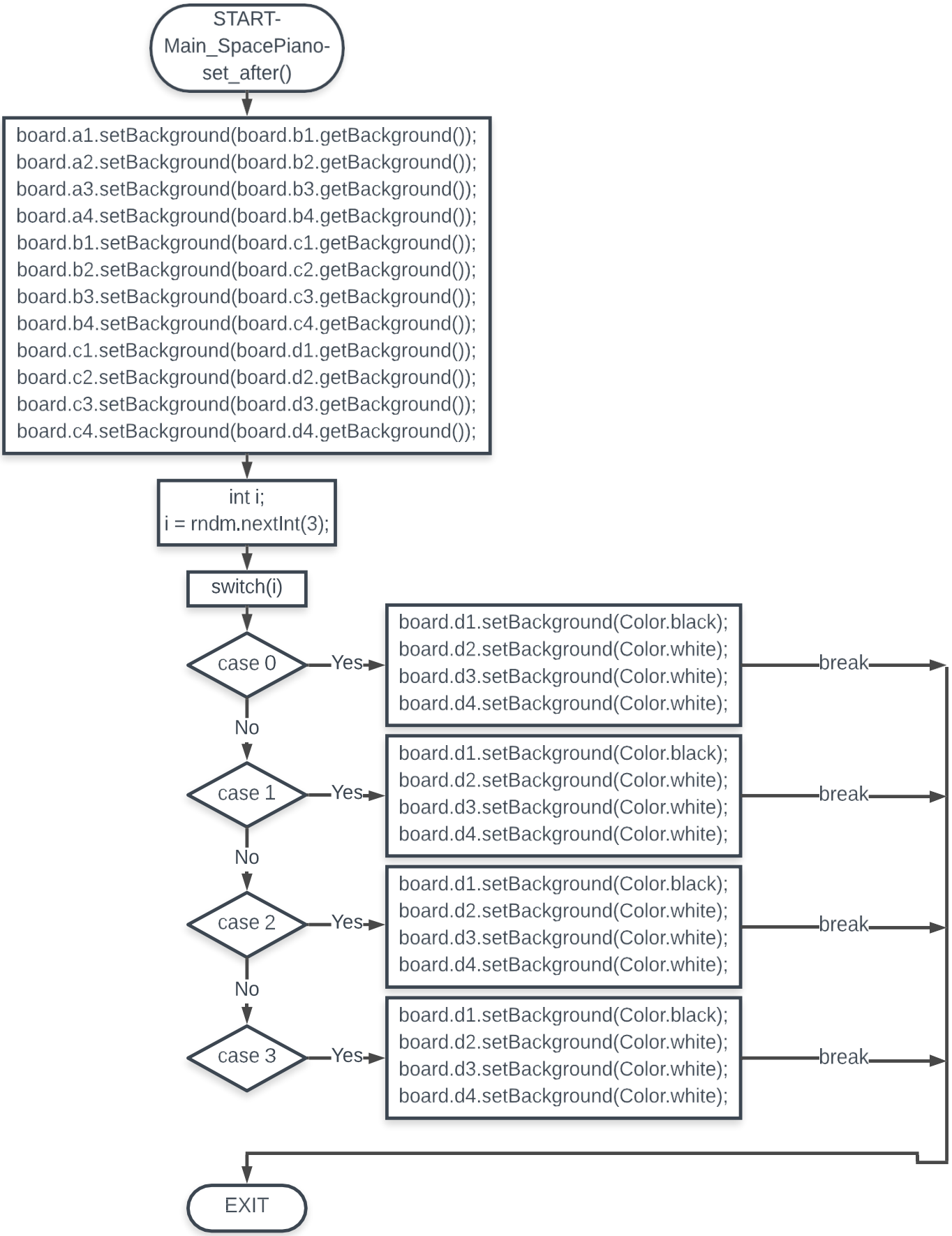

if (score > 399 && score <= 480)
Yes
count = 1500;
No
if (score > 480)
Yes
count = 1200;
No
board.timer.restart();
if (board.a1.getParent().getBackground().equals(Color.red) || board.a2.getParent().getBackground().equals(Color.red) || board.a3.getParent().getBackground().equals(Color.red) || board.a4.getParent().getBackground().equals(Color.red) || board.t2.getText().equals("Out of Time"))
Yes
2337 MILLISECONDS
board.timer.stop();
board.dispose();
Object Creation
SP_Scores b = new SP_Scores();
if (board.t2.getText().equals("Out of Time"))
Yes
No

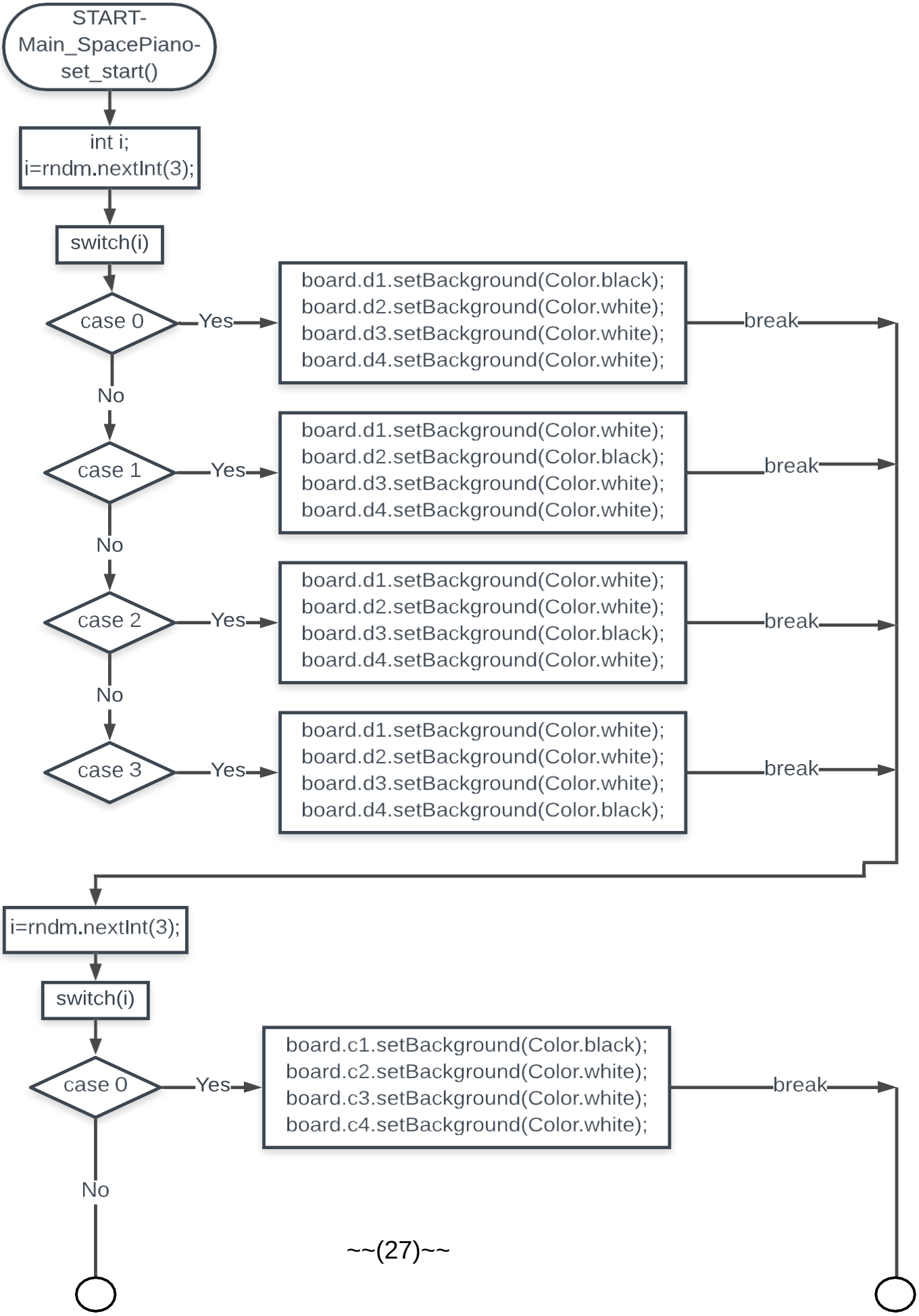

l1 = ("Time Out");
l1 = ("You have clicked wrong tile");
l3 = (Long.toString(score));
Try block
=ileInputStream fis = new
FileInputStream("ab.txt");
ObjectInputStream ois = new
ObjectInputStream(fis);
highest = (long[]) ois.readObject();
ois.close();
Exception
Occurs
true
Catch
block
false
if (highest[0] < score)
Yes
No
t1 = ("New highest scores.");
highest[0] = score;
Try block
=ileOutputStream fos = new
=ileOutputStream("ab.txt");
ObjectOutputStream oos = new
ObjectOutputStream(fos);
oos.writeObject(highest);
oos.close();
Exception
Occurs
true
Catch
block
false

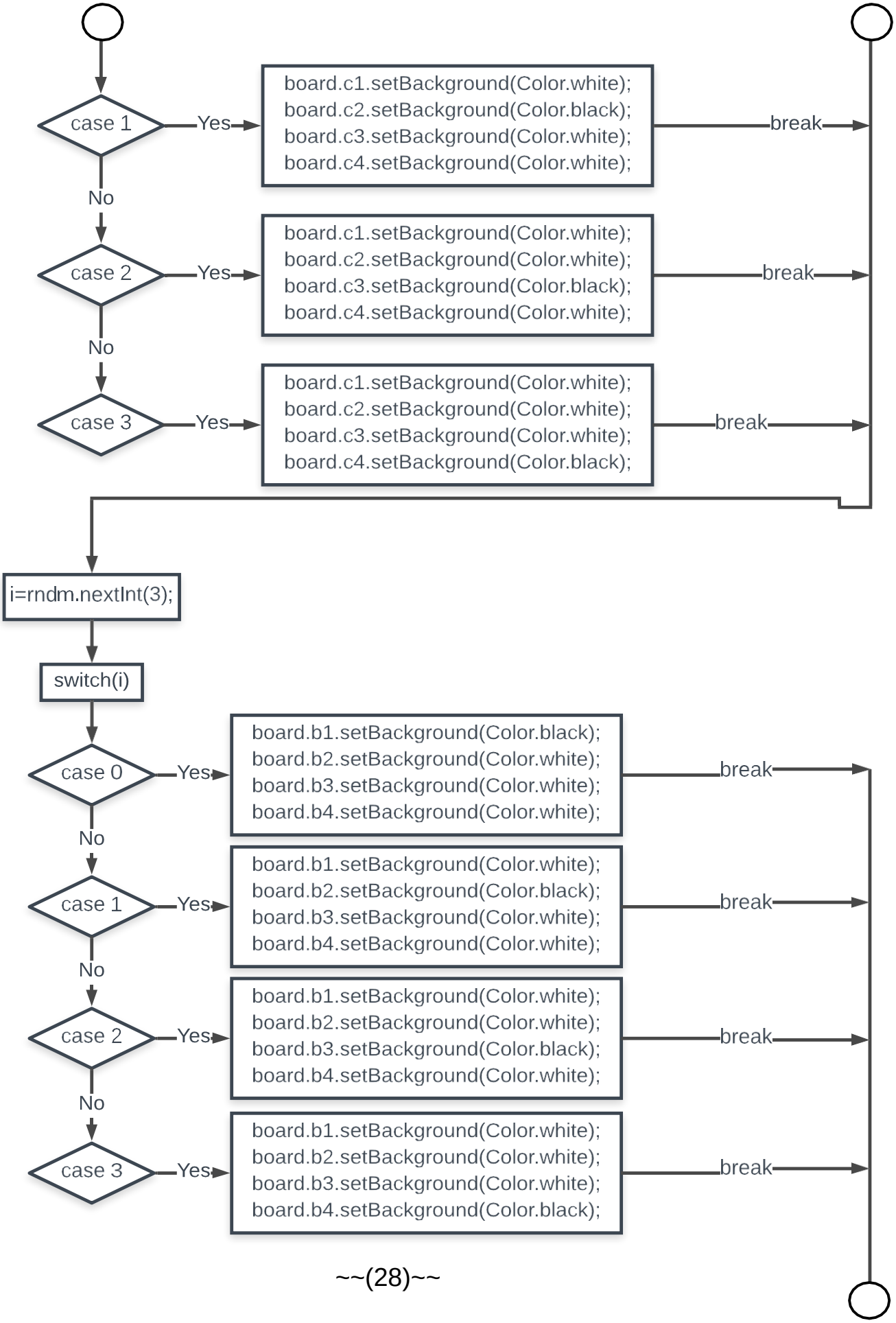

if (highest[0] > score)
Yes
No
t1 = ("Highest Scores are :- ");
t2 = (Long.toString(highest[0]));
if (highest[0] == score)
Yes
No
t1 = ("Scores similar to −ighest Scores");
Main.exit_a = true;
EXIT

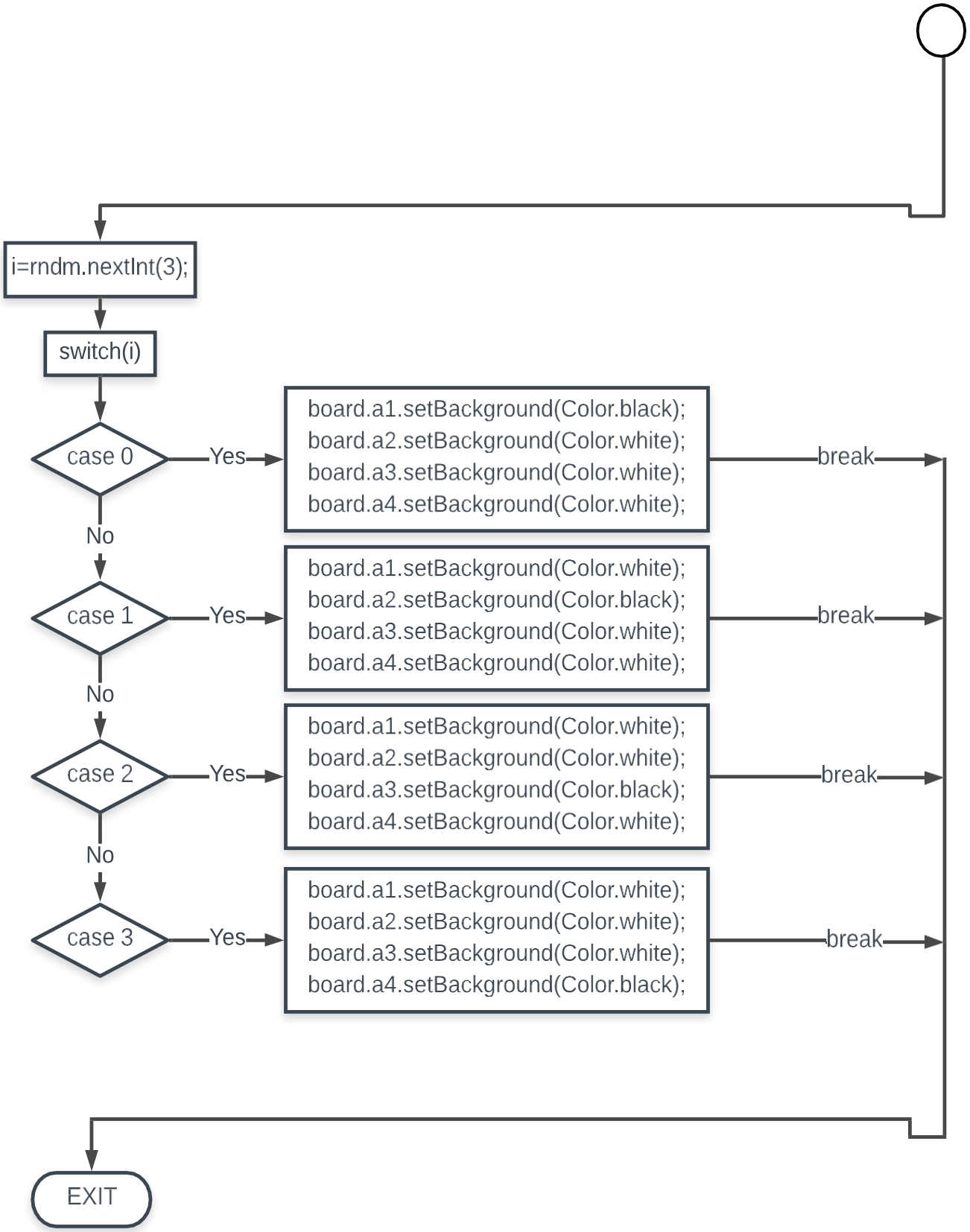

START-
Main_SpacePiano-
main()
Class members declaration and initialisation
public static boolean correct = false;
private static SpacePiano board = new SpacePiano();
private static Scanner sc = new Scanner(System.in);
private static Random rndm = new Random();
private static TimeUnit Unit = TimeUnit.MILLISECONDS;
private static long[] highest = {0};
public static int count = 3000;
public static String l1 = "", l3 = "", t1 = "", t2 = "";
private static long score = -1;
score = -1;
board.setBackground(Color.white);
board.a1.getParent().setBackground(Color.white);
board.a2.getParent().setBackground(Color.white);
board.a3.getParent().setBackground(Color.white);
board.a4.getParent().setBackground(Color.white);
if (board.isVisible() == false)
true
Main_SpacePiano-
setstart();
board.setVisible(true);
board.l2.setText("0");
count = 3000;
board.t2.setText("Start");
board.toFront();
false
EXIT

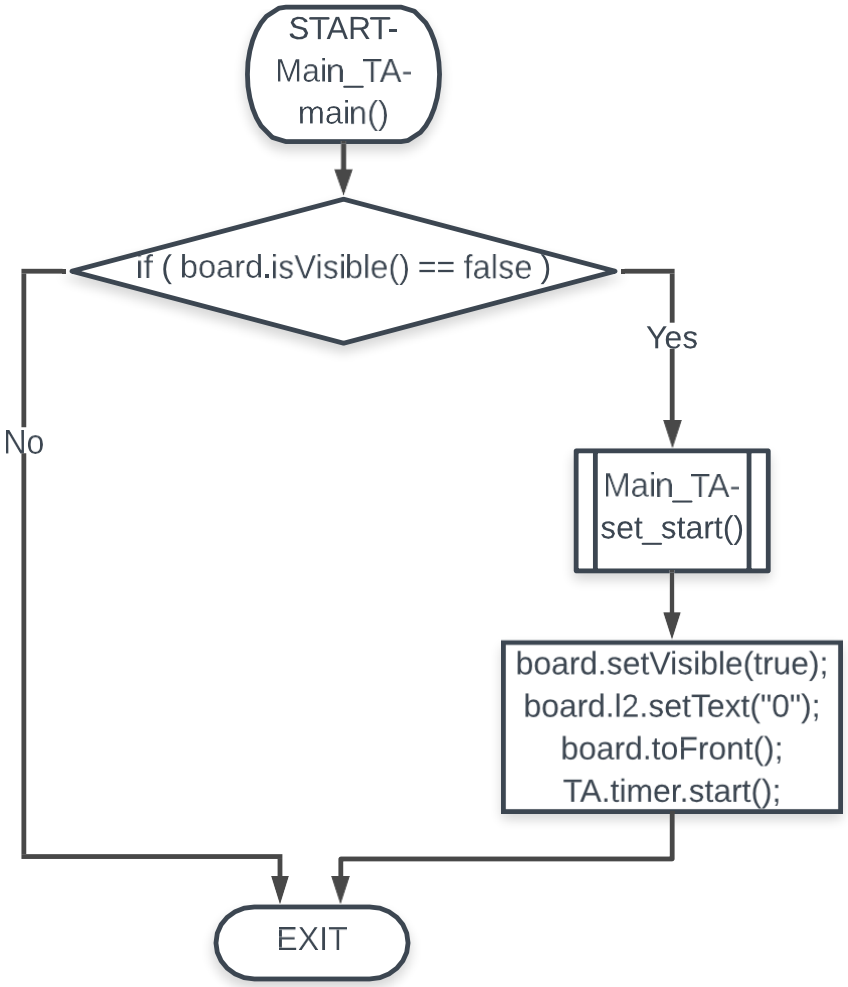

START-
Main_SpacePiano-
time()
count -= 2;
if (count == 0)
true
board.t2.setText("Out of Time");
false
if (count > 0)
true
int misec = count % 1000;
int sec = count / 1000;
int min = sec / 60;
board.t2.setText(min + " : " + sec + " . " + misec);
false
EXIT

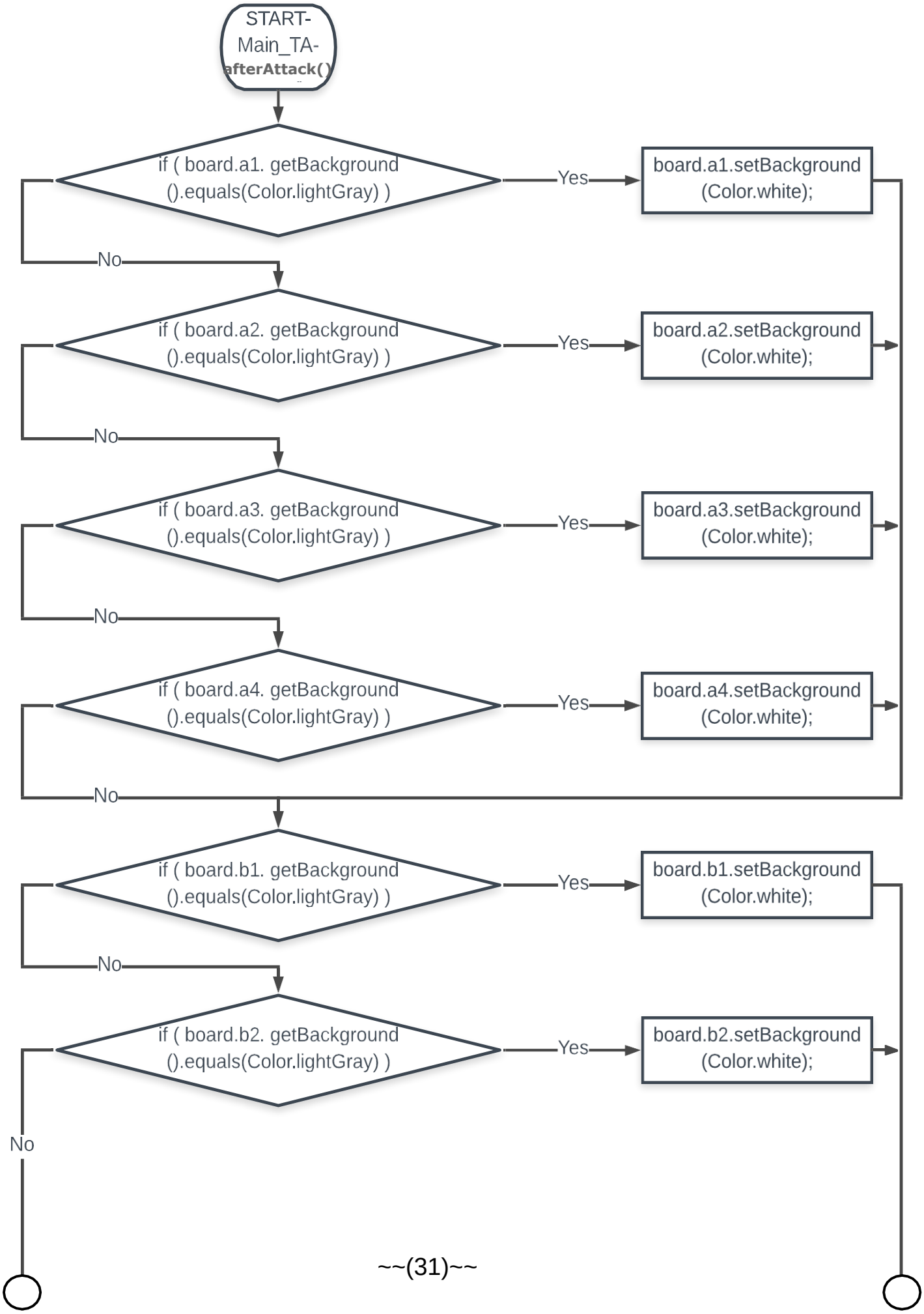

START-
Main_SpacePiano-
set_after()
board.a1.setBackground(board.b1.getBackground());
board.a2.setBackground(board.b2.getBackground());
board.a3.setBackground(board.b3.getBackground());
board.a4.setBackground(board.b4.getBackground());
board.b1.setBackground(board.c1.getBackground());
board.b2.setBackground(board.c2.getBackground());
board.b3.setBackground(board.c3.getBackground());
board.b4.setBackground(board.c4.getBackground());
board.c1.setBackground(board.d1.getBackground());
board.c2.setBackground(board.d2.getBackground());
board.c3.setBackground(board.d3.getBackground());
board.c4.setBackground(board.d4.getBackground());
int i;
i = rndm.nextInt(3);
switch(i)
case 0
Yes
board.d1.setBackground(Color.black);
board.d2.setBackground(Color.white);
board.d3.setBackground(Color.white);
board.d4.setBackground(Color.white);
break
No
case 1
Yes
board.d1.setBackground(Color.black);
board.d2.setBackground(Color.white);
board.d3.setBackground(Color.white);
board.d4.setBackground(Color.white);
break
No
case 2
Yes
board.d1.setBackground(Color.black);
board.d2.setBackground(Color.white);
board.d3.setBackground(Color.white);
board.d4.setBackground(Color.white);
break
No
case 3
Yes
board.d1.setBackground(Color.black);
board.d2.setBackground(Color.white);
board.d3.setBackground(Color.white);
board.d4.setBackground(Color.white);
break
EXIT

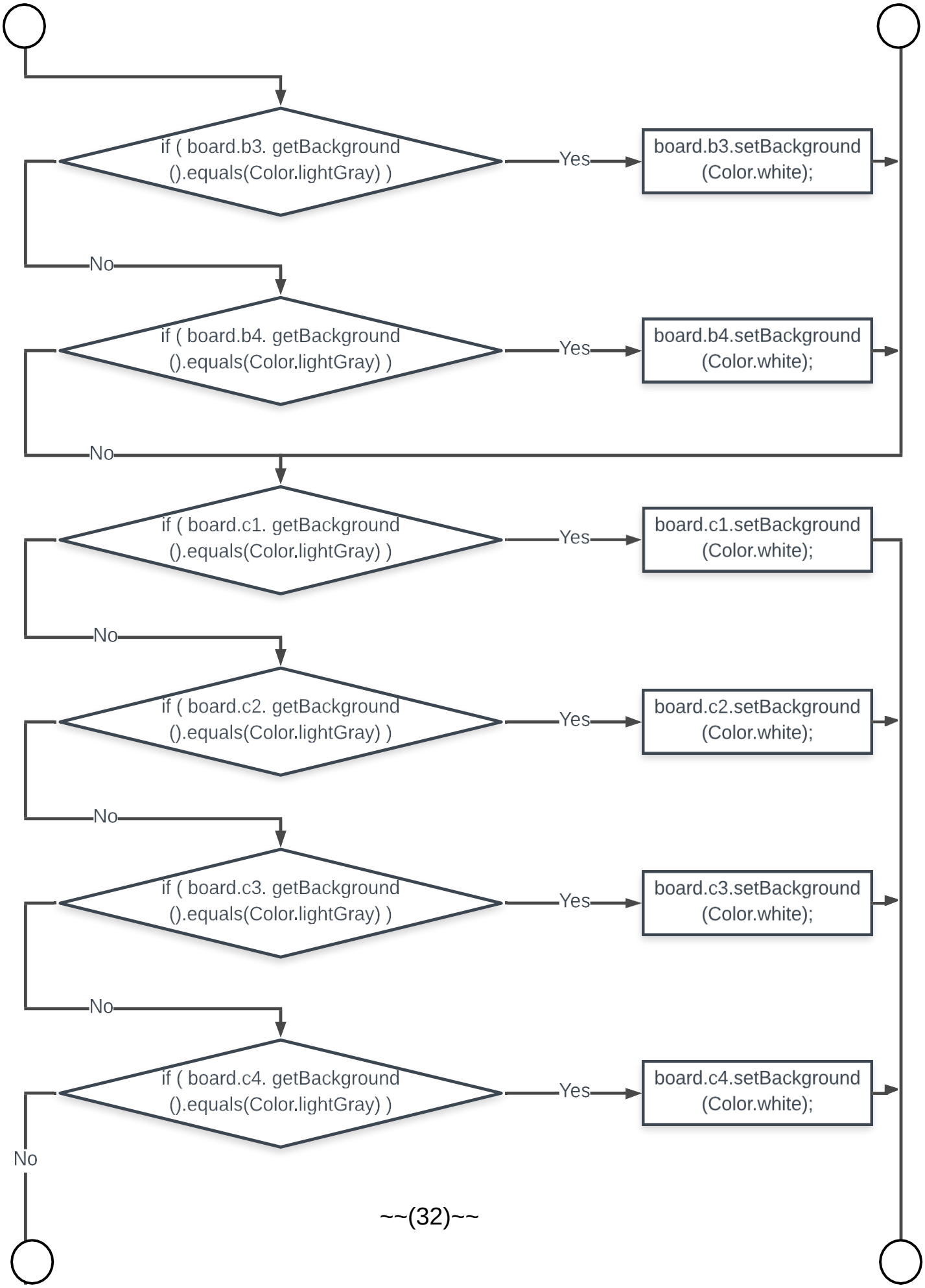

START-
Main_SpacePiano-
set_start()
int i;
i=rndm.nextInt(3);
switch(i)
case 0
Yes
board.d1.setBackground(Color.black);
board.d2.setBackground(Color.white);
board.d3.setBackground(Color.white);
board.d4.setBackground(Color.white);
break
No
case 1
Yes
board.d1.setBackground(Color.white);
board.d2.setBackground(Color.black);
board.d3.setBackground(Color.white);
board.d4.setBackground(Color.white);
break
No
case 2
Yes
board.d1.setBackground(Color.white);
board.d2.setBackground(Color.white);
board.d3.setBackground(Color.black);
board.d4.setBackground(Color.white);
break
No
case 3
Yes
board.d1.setBackground(Color.white);
board.d2.setBackground(Color.white);
board.d3.setBackground(Color.white);
board.d4.setBackground(Color.black);
break
i=rndm.nextInt(3);
switch(i)
case 0
Yes
board.c1.setBackground(Color.black);
board.c2.setBackground(Color.white);
board.c3.setBackground(Color.white);
board.c4.setBackground(Color.white);
break
No

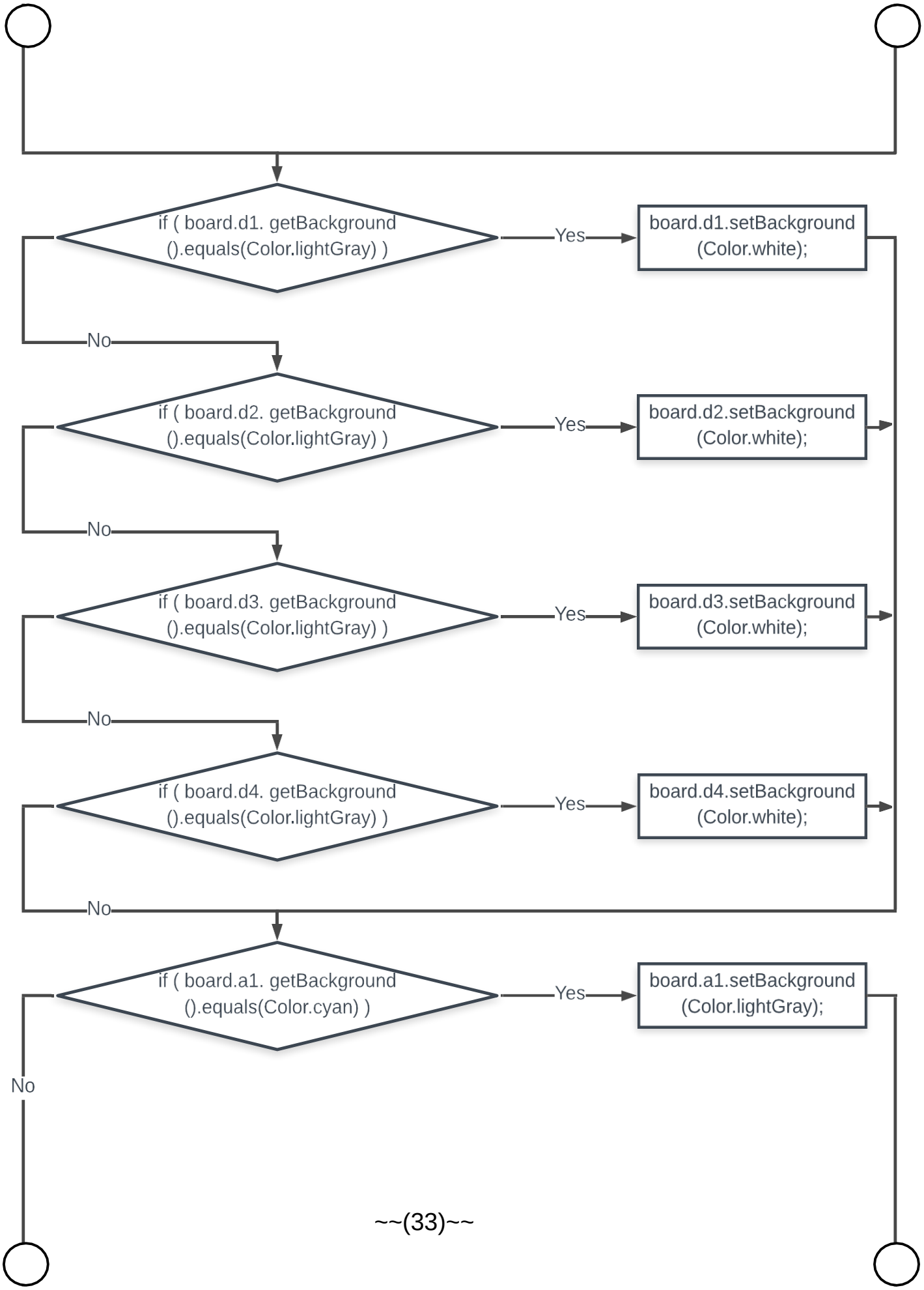

case 1
Yes
board.c1.setBackground(Color.white);
board.c2.setBackground(Color.black);
board.c3.setBackground(Color.white);
board.c4.setBackground(Color.white);
break
No
case 2
Yes
board.c1.setBackground(Color.white);
board.c2.setBackground(Color.white);
board.c3.setBackground(Color.black);
board.c4.setBackground(Color.white);
break
No
case 3
Yes
board.c1.setBackground(Color.white);
board.c2.setBackground(Color.white);
board.c3.setBackground(Color.white);
board.c4.setBackground(Color.black);
break
i=rndm.nextInt(3);
switch(i)
case 0
Yes
board.b1.setBackground(Color.black);
board.b2.setBackground(Color.white);
board.b3.setBackground(Color.white);
board.b4.setBackground(Color.white);
break
No
case 1
Yes
board.b1.setBackground(Color.white);
board.b2.setBackground(Color.black);
board.b3.setBackground(Color.white);
board.b4.setBackground(Color.white);
break
No
case 2
Yes
board.b1.setBackground(Color.white);
board.b2.setBackground(Color.white);
board.b3.setBackground(Color.black);
board.b4.setBackground(Color.white);
break
No
case 3
Yes
board.b1.setBackground(Color.white);
board.b2.setBackground(Color.white);
board.b3.setBackground(Color.white);
board.b4.setBackground(Color.black);
break

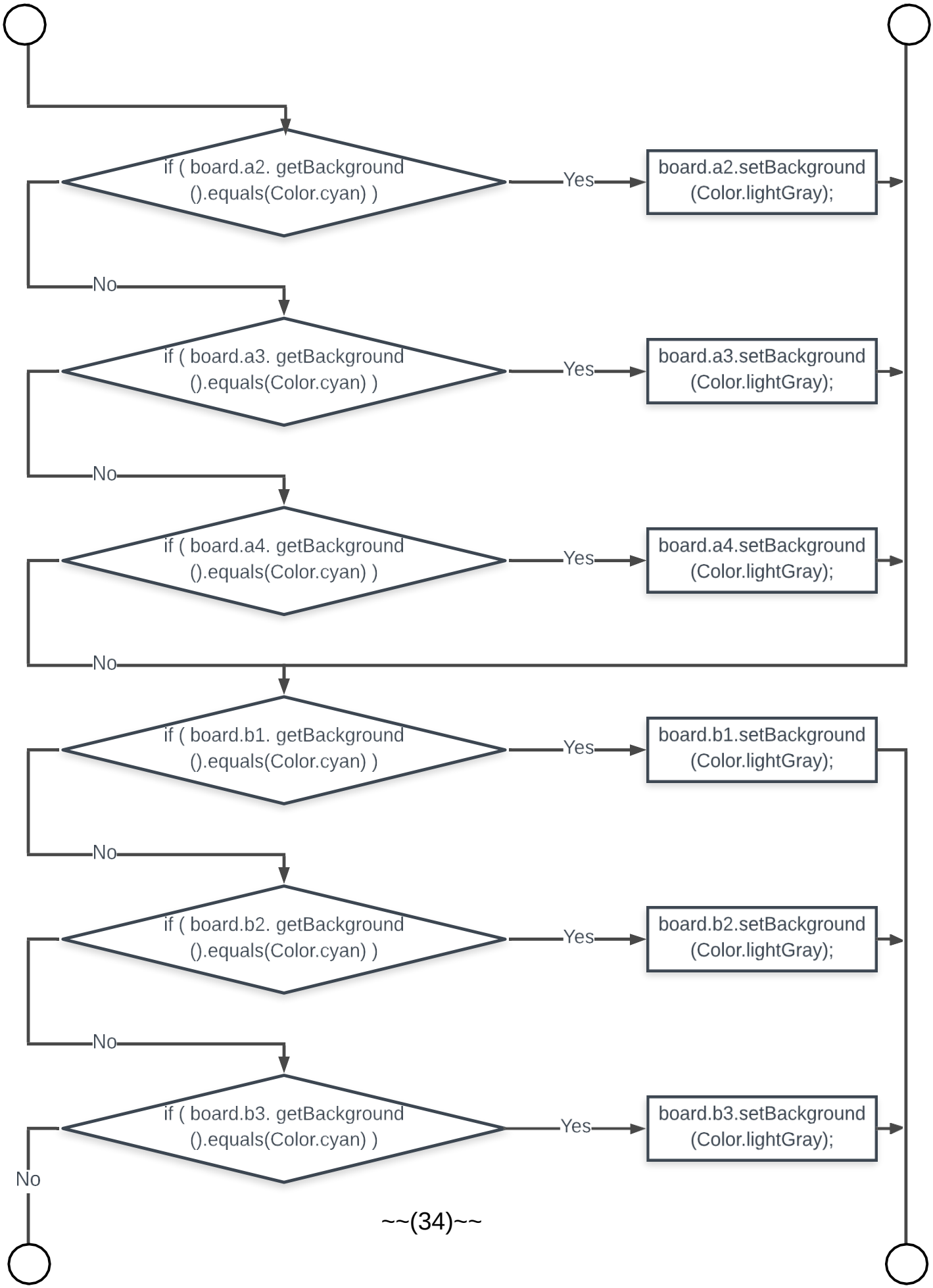

i=rndm.nextInt(3);
switch(i)
case 0
Yes
board.a1.setBackground(Color.black);
board.a2.setBackground(Color.white);
board.a3.setBackground(Color.white);
board.a4.setBackground(Color.white);
break
No
case 1
Yes
board.a1.setBackground(Color.white);
board.a2.setBackground(Color.black);
board.a3.setBackground(Color.white);
board.a4.setBackground(Color.white);
break
No
case 2
Yes
board.a1.setBackground(Color.white);
board.a2.setBackground(Color.white);
board.a3.setBackground(Color.black);
board.a4.setBackground(Color.white);
break
No
case 3
Yes
board.a1.setBackground(Color.white);
board.a2.setBackground(Color.white);
board.a3.setBackground(Color.white);
board.a4.setBackground(Color.black);
break
EXIT

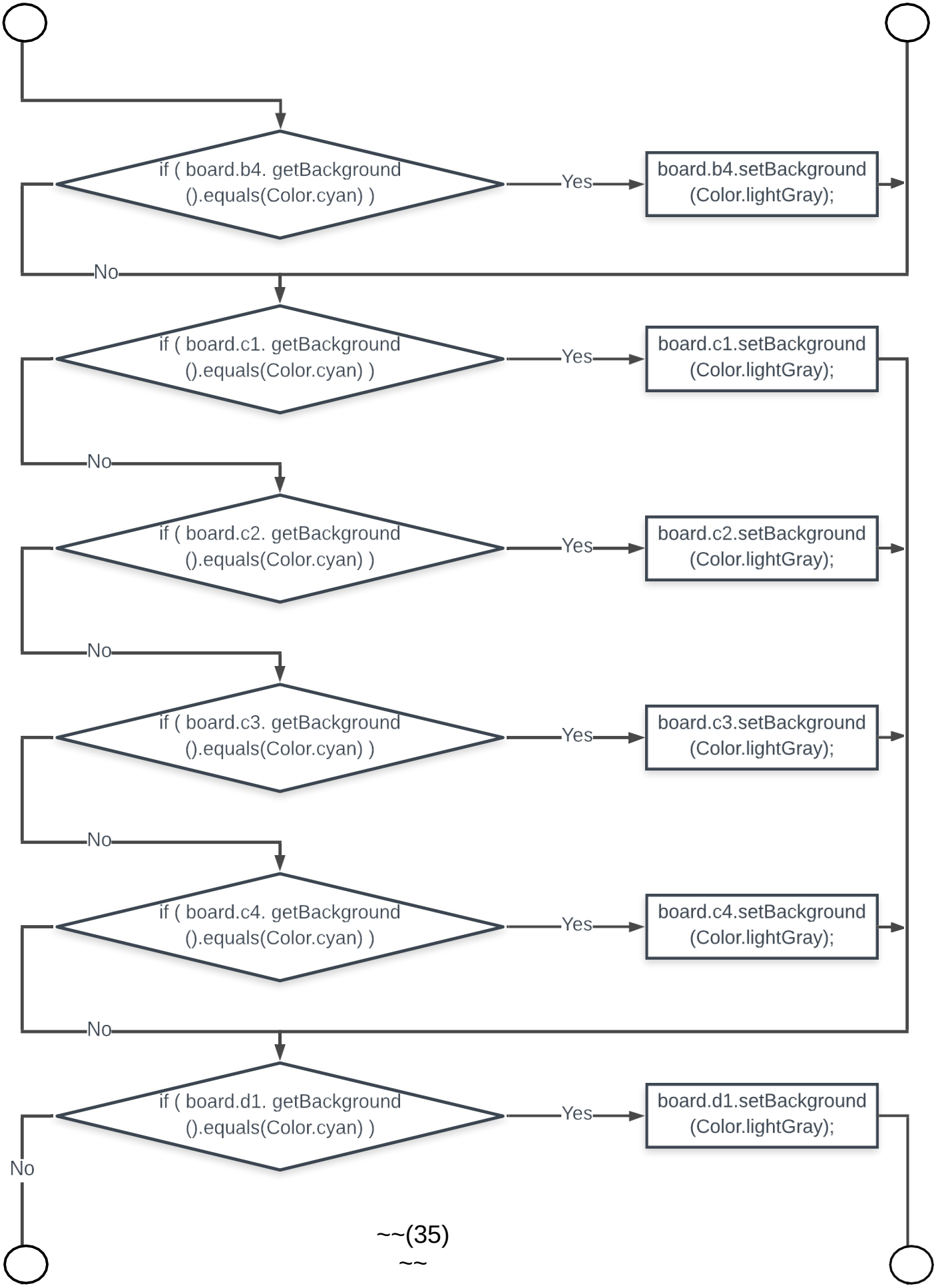

START-
Main_TA-
main()
if (board.isVisible() == false)
Yes
No
Main_TA-
set_start()
board.setVisible(true);
board.l2.setText("0");
board.toFront();
TA.timer.start();
EXIT

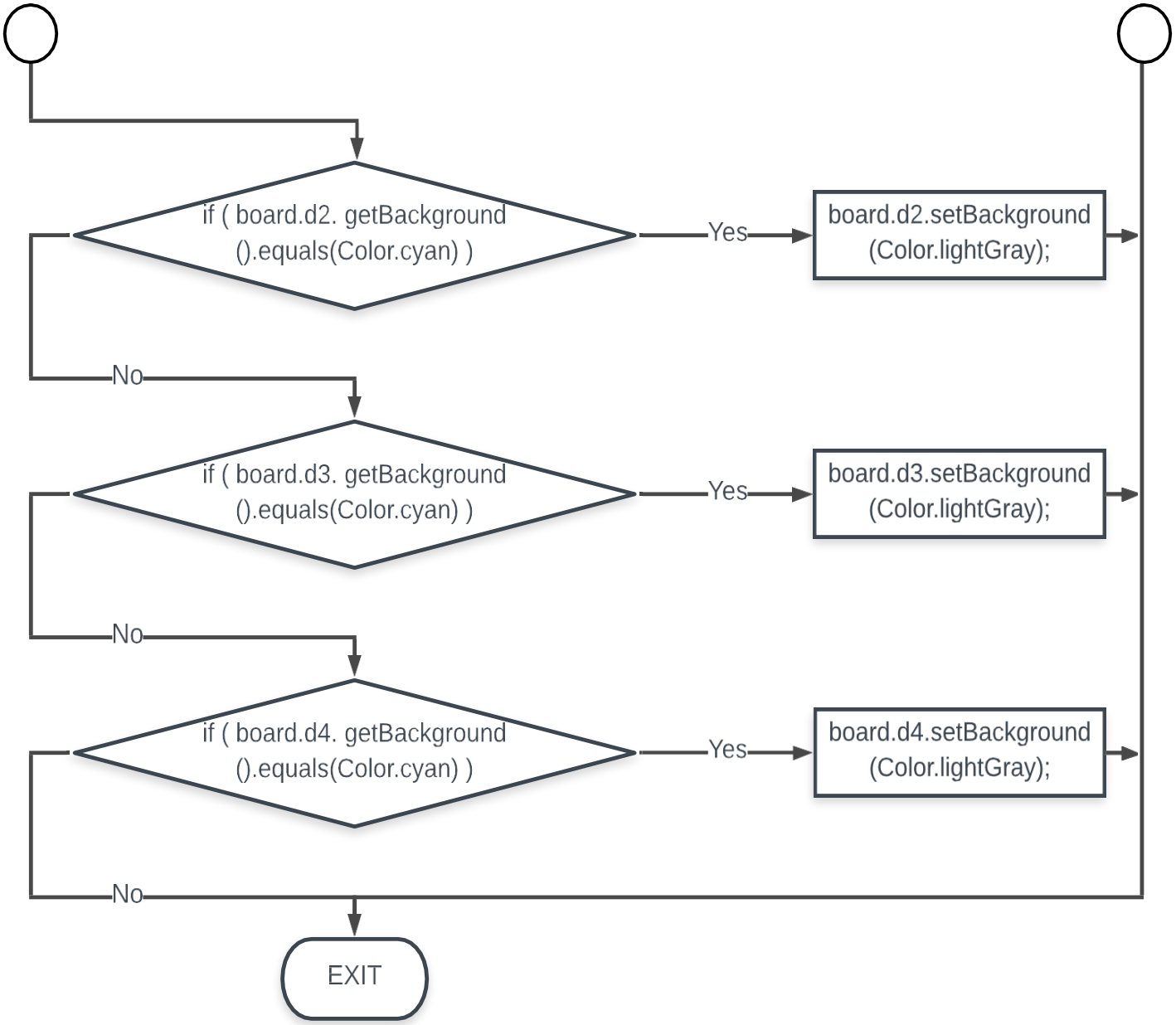

START-
Main_TA-
afterAttack()
if (board.a1. getBackground
().equals(Color.lightGray))
Yes
board.a1.setBackground
(Color.white);
No
if (board.a2. getBackground
().equals(Color.lightGray))
Yes
board.a2.setBackground
(Color.white);
No
if (board.a3. getBackground
().equals(Color.lightGray))
Yes
board.a3.setBackground
(Color.white);
No
if (board.a4. getBackground
().equals(Color.lightGray))
Yes
board.a4.setBackground
(Color.white);
No
if (board.b1. getBackground
().equals(Color.lightGray))
Yes
board.b1.setBackground
(Color.white);
No
if (board.b2. getBackground
().equals(Color.lightGray))
Yes
board.b2.setBackground
(Color.white);
No

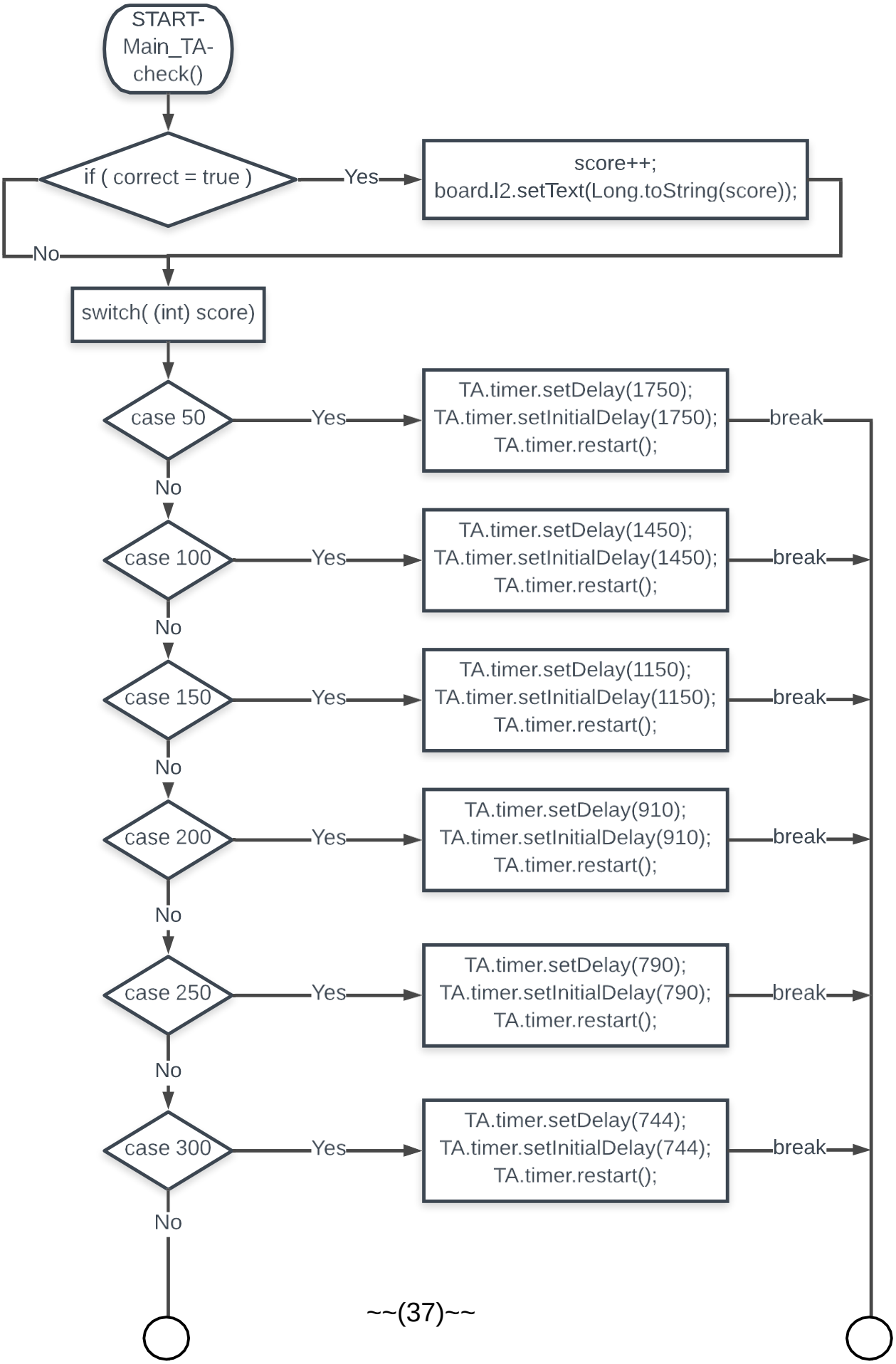

if (board.b3. getBackground ().equals(Color.lightGray))
Yes
board.b3.setBackground (Color.white);
No
if (board.b4. getBackground ().equals(Color.lightGray))
Yes
board.b4.setBackground (Color.white);
No
if (board.c1. getBackground ().equals(Color.lightGray))
Yes
board.c1.setBackground (Color.white);
No
if (board.c2. getBackground ().equals(Color.lightGray))
Yes
board.c2.setBackground (Color.white);
No
if (board.c3. getBackground ().equals(Color.lightGray))
Yes
board.c3.setBackground (Color.white);
No
if (board.c4. getBackground ().equals(Color.lightGray))
Yes
board.c4.setBackground (Color.white);
No

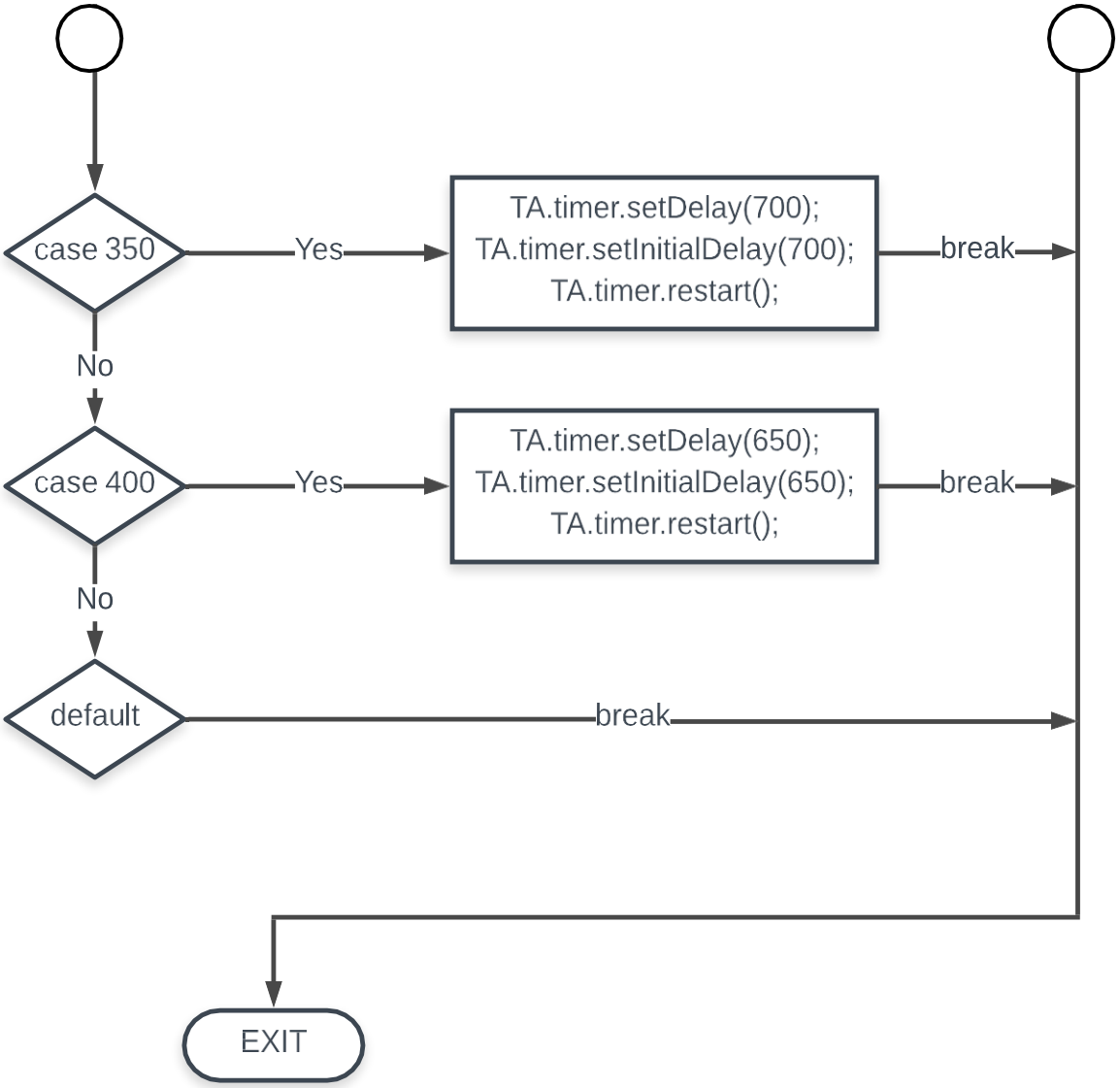

if (board.d1. getBackground ().equals(Color.lightGray))
Yes
board.d1.setBackground (Color.white);
No
if (board.d2. getBackground ().equals(Color.lightGray))
Yes
board.d2.setBackground (Color.white);
No
if (board.d3. getBackground ().equals(Color.lightGray))
Yes
board.d3.setBackground (Color.white);
No
if (board.d4. getBackground ().equals(Color.lightGray))
Yes
board.d4.setBackground (Color.white);
No
if (board.a1. getBackground ().equals(Color.cyan))
Yes
board.a1.setBackground (Color.lightGray);
No

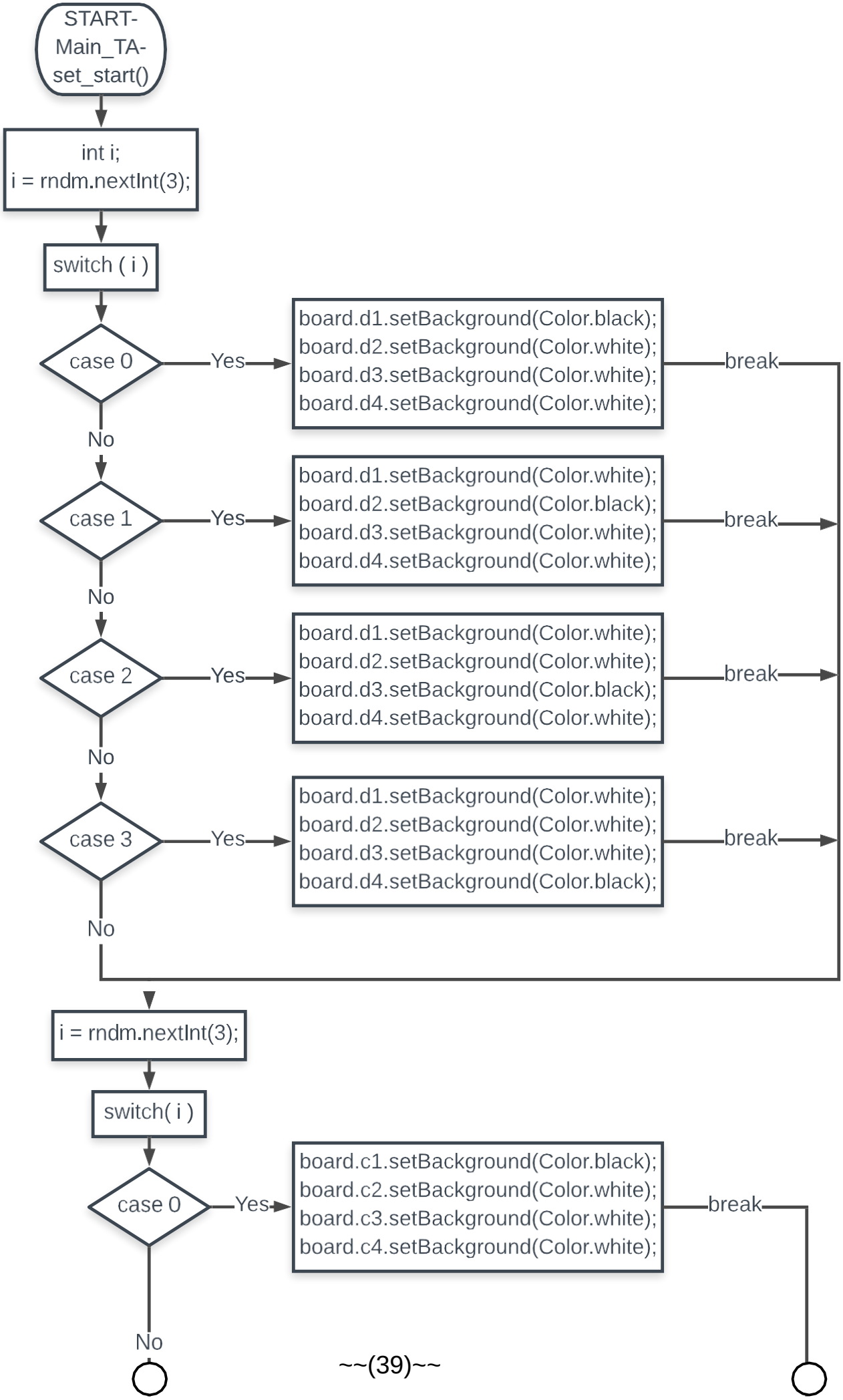

if (board.a2. getBackground ().equals(Color.cyan))
Yes
board.a2.setBackground (Color.lightGray);
No
if (board.a3. getBackground ().equals(Color.cyan))
Yes
board.a3.setBackground (Color.lightGray);
No
if (board.a4. getBackground ().equals(Color.cyan))
Yes
board.a4.setBackground (Color.lightGray);
No
if (board.b1. getBackground ().equals(Color.cyan))
Yes
board.b1.setBackground (Color.lightGray);
No
if (board.b2. getBackground ().equals(Color.cyan))
Yes
board.b2.setBackground (Color.lightGray);
No
if (board.b3. getBackground ().equals(Color.cyan))
Yes
board.b3.setBackground (Color.lightGray);
No

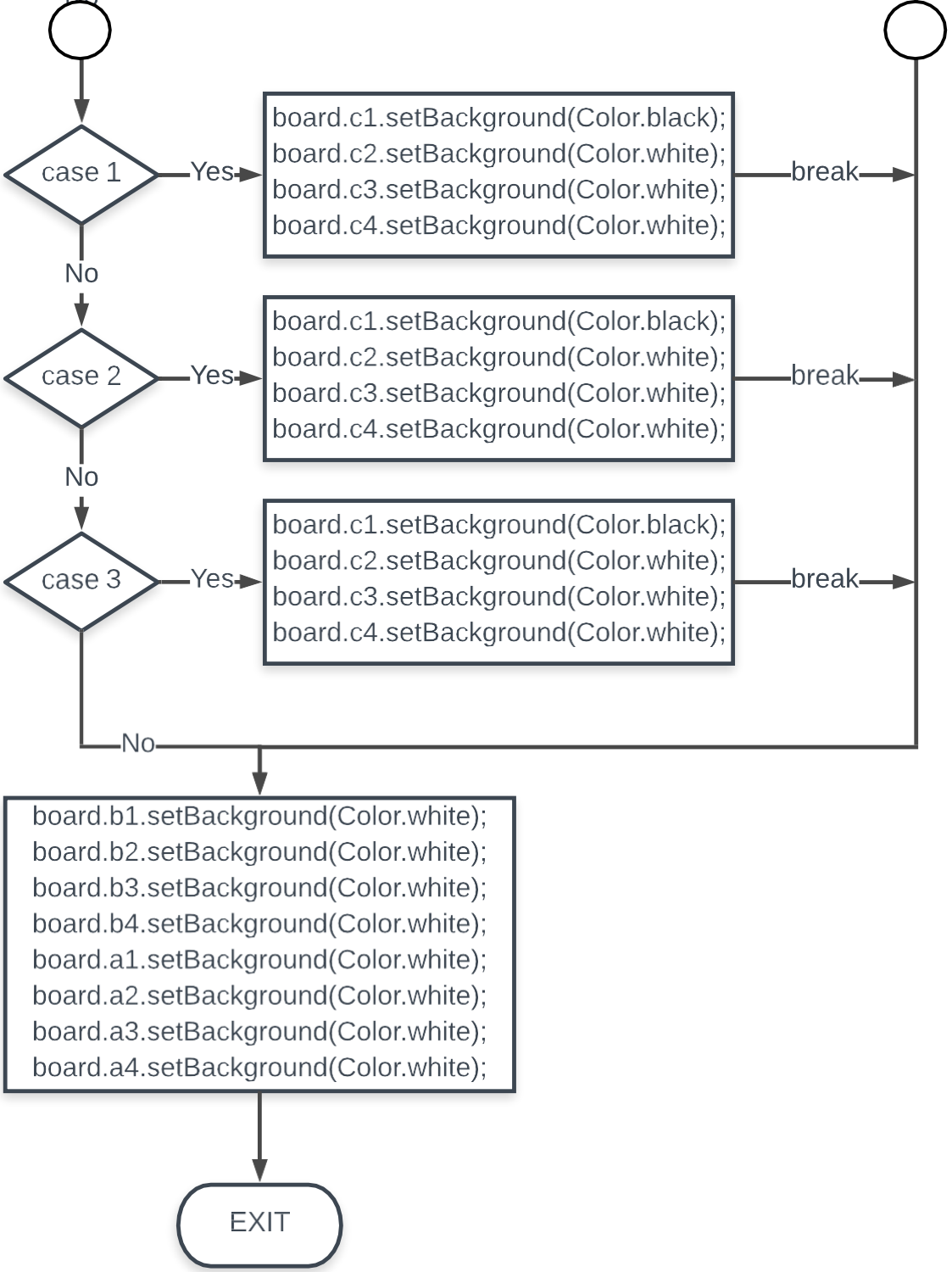

if (board.b4. getBackground ().equals(Color.cyan))
Yes
board.b4.setBackground (Color.lightGray);
No
if (board.c1. getBackground ().equals(Color.cyan))
Yes
board.c1.setBackground (Color.lightGray);
No
if (board.c2. getBackground ().equals(Color.cyan))
Yes
board.c2.setBackground (Color.lightGray);
No
if (board.c3. getBackground ().equals(Color.cyan))
Yes
board.c3.setBackground (Color.lightGray);
No
if (board.c4. getBackground ().equals(Color.cyan))
Yes
board.c4.setBackground (Color.lightGray);
No
if (board.d1. getBackground ().equals(Color.cyan))
Yes
board.d1.setBackground (Color.lightGray);
No

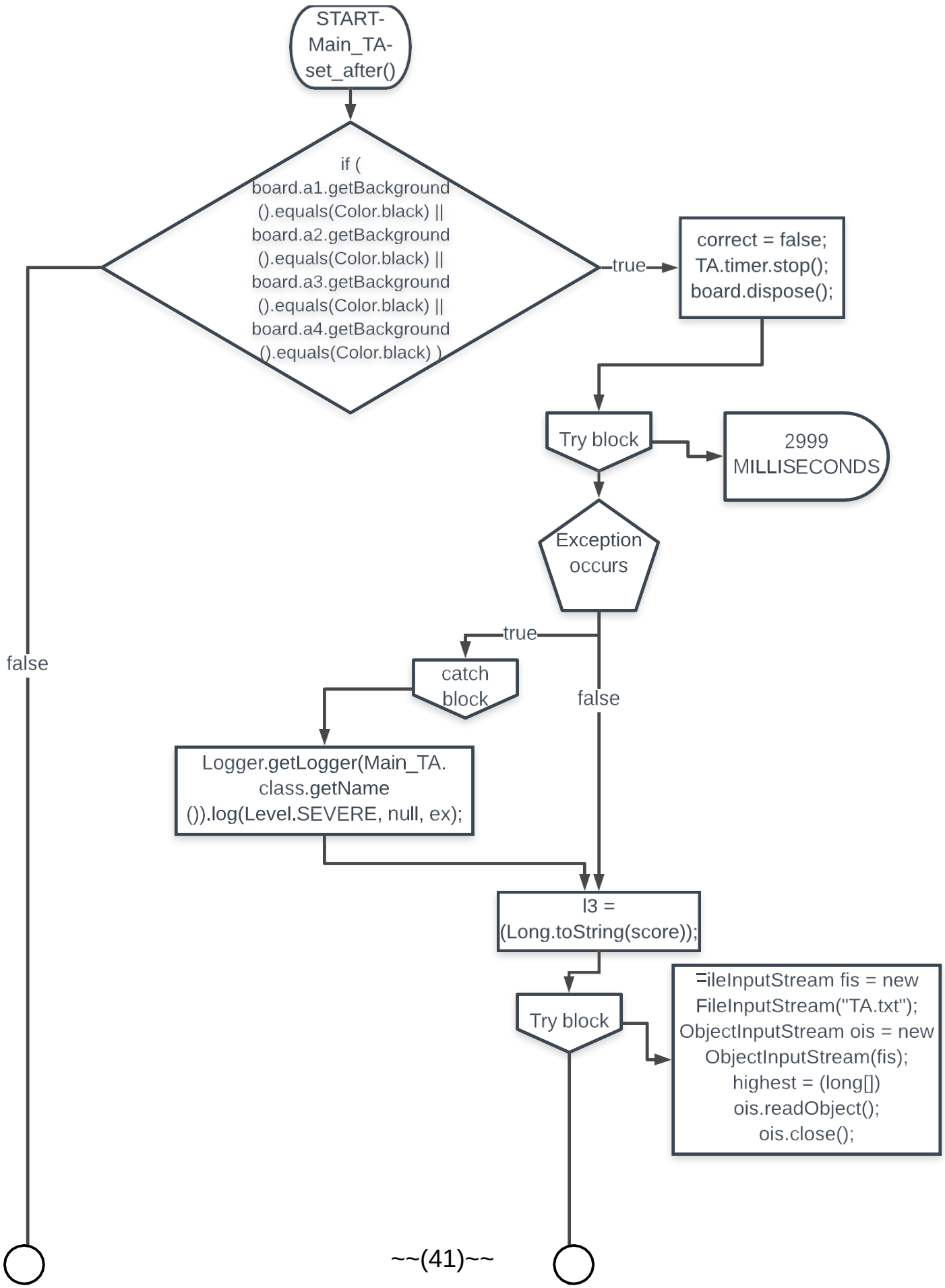

if (board.d2. getBackground ().equals(Color.cyan))
Yes
board.d2.setBackground (Color.lightGray);
No
if (board.d3. getBackground ().equals(Color.cyan))
Yes
board.d3.setBackground (Color.lightGray);
No
if (board.d4. getBackground ().equals(Color.cyan))
Yes
board.d4.setBackground (Color.lightGray);
No
EXIT

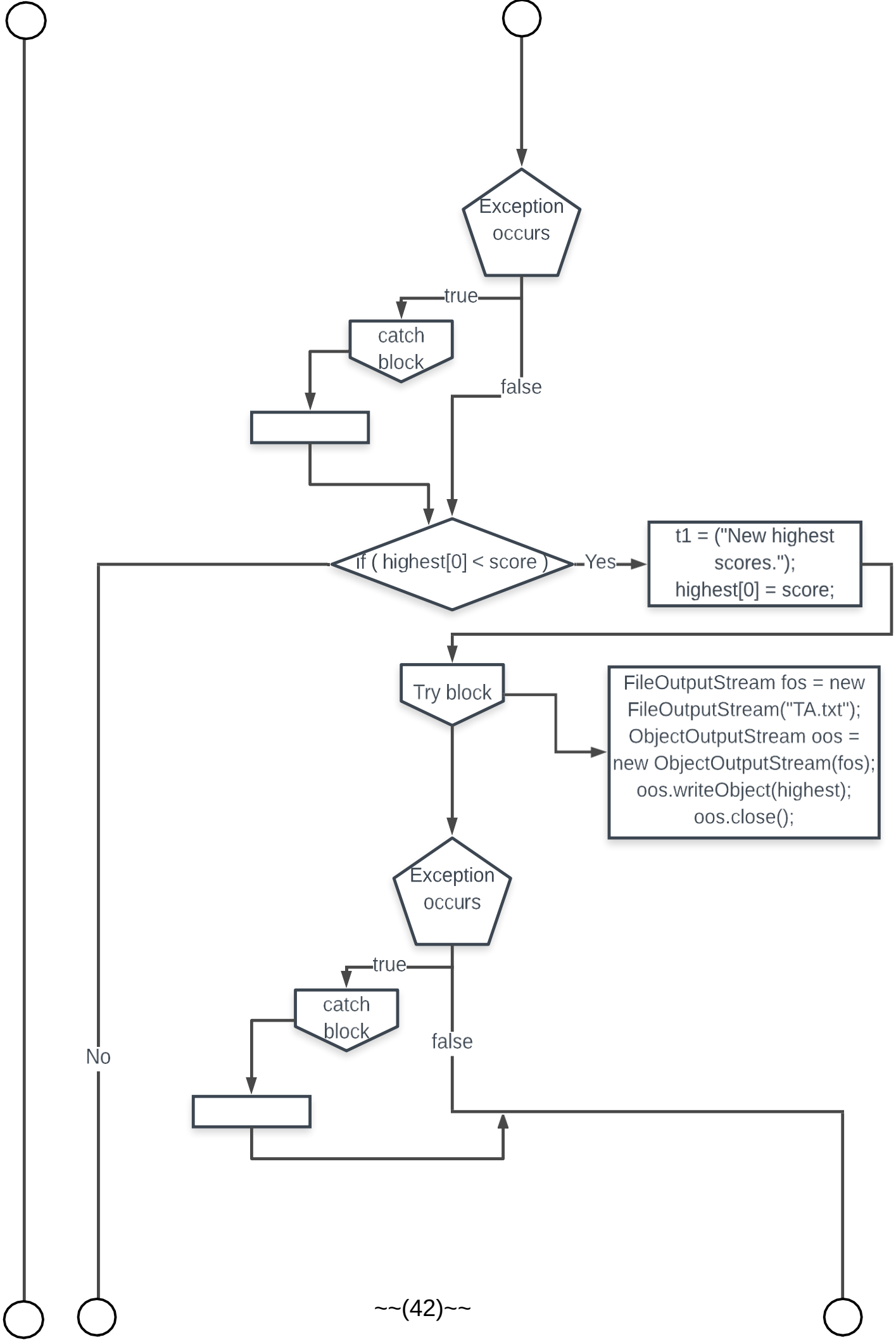

START-
Main_TA-
check()
if (correct = true)
Yes
score++;
board.l2.setText(Long.toString(score));
No
switch((int) score)
case 50
Yes
TA.timer.setDelay(1750);
TA.timer.setInitialDelay(1750);
TA.timer.restart();
break
No
case 100
Yes
TA.timer.setDelay(1450);
TA.timer.setInitialDelay(1450);
TA.timer.restart();
break
No
case 150
Yes
TA.timer.setDelay(1150);
TA.timer.setInitialDelay(1150);
TA.timer.restart();
break
No
case 200
Yes
TA.timer.setDelay(910);
TA.timer.setInitialDelay(910);
TA.timer.restart();
break
No
case 250
Yes
TA.timer.setDelay(790);
TA.timer.setInitialDelay(790);
TA.timer.restart();
break
No
case 300
Yes
TA.timer.setDelay(744);
TA.timer.setInitialDelay(744);
TA.timer.restart();
break
No

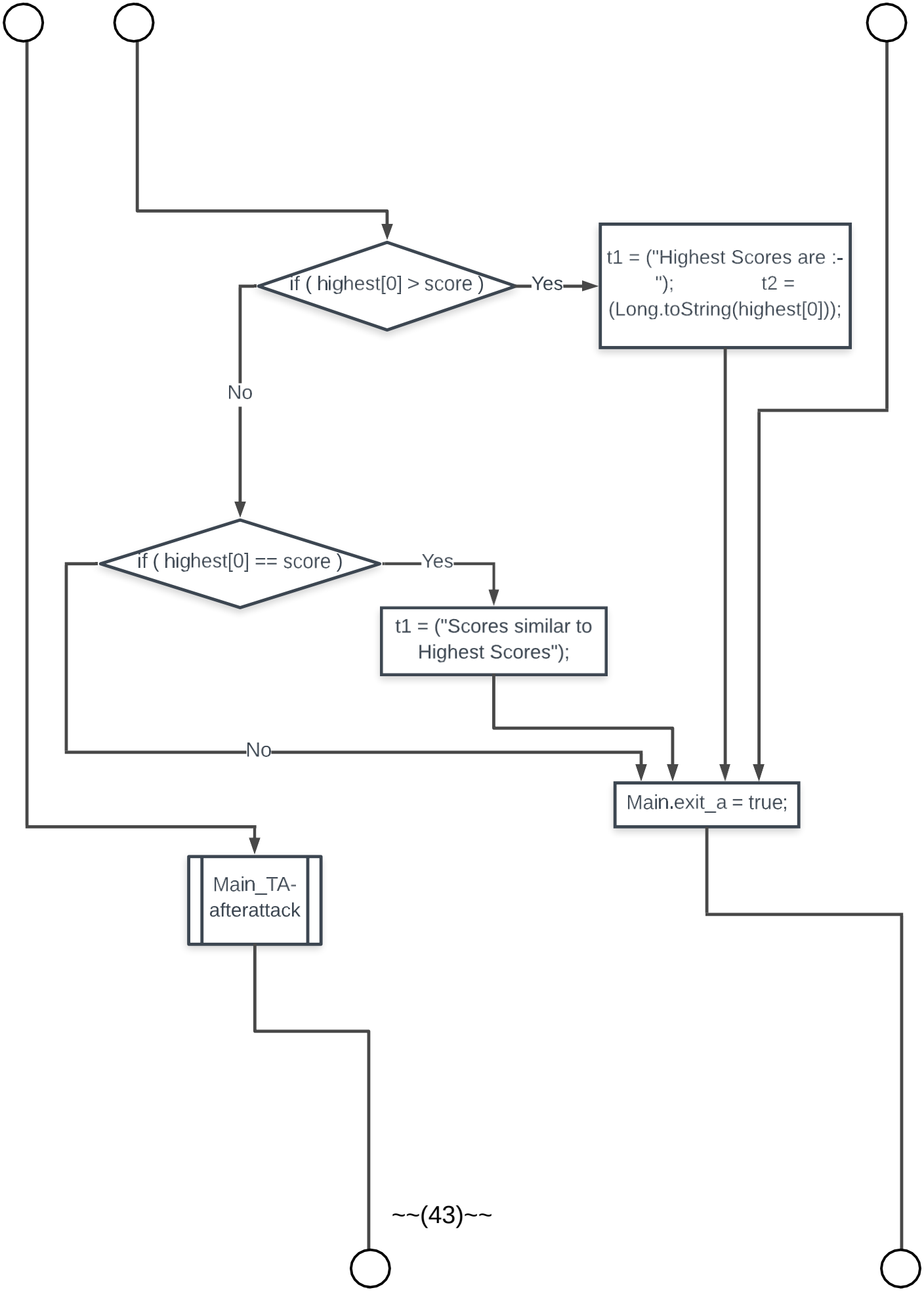

case 350
Yes
TA.timer.setDelay(700);
TA.timer.setInitialDelay(700);
TA.timer.restart();
break
No
case 400
Yes
TA.timer.setDelay(650);
TA.timer.setInitialDelay(650);
TA.timer.restart();
break
No
default
break
EXIT

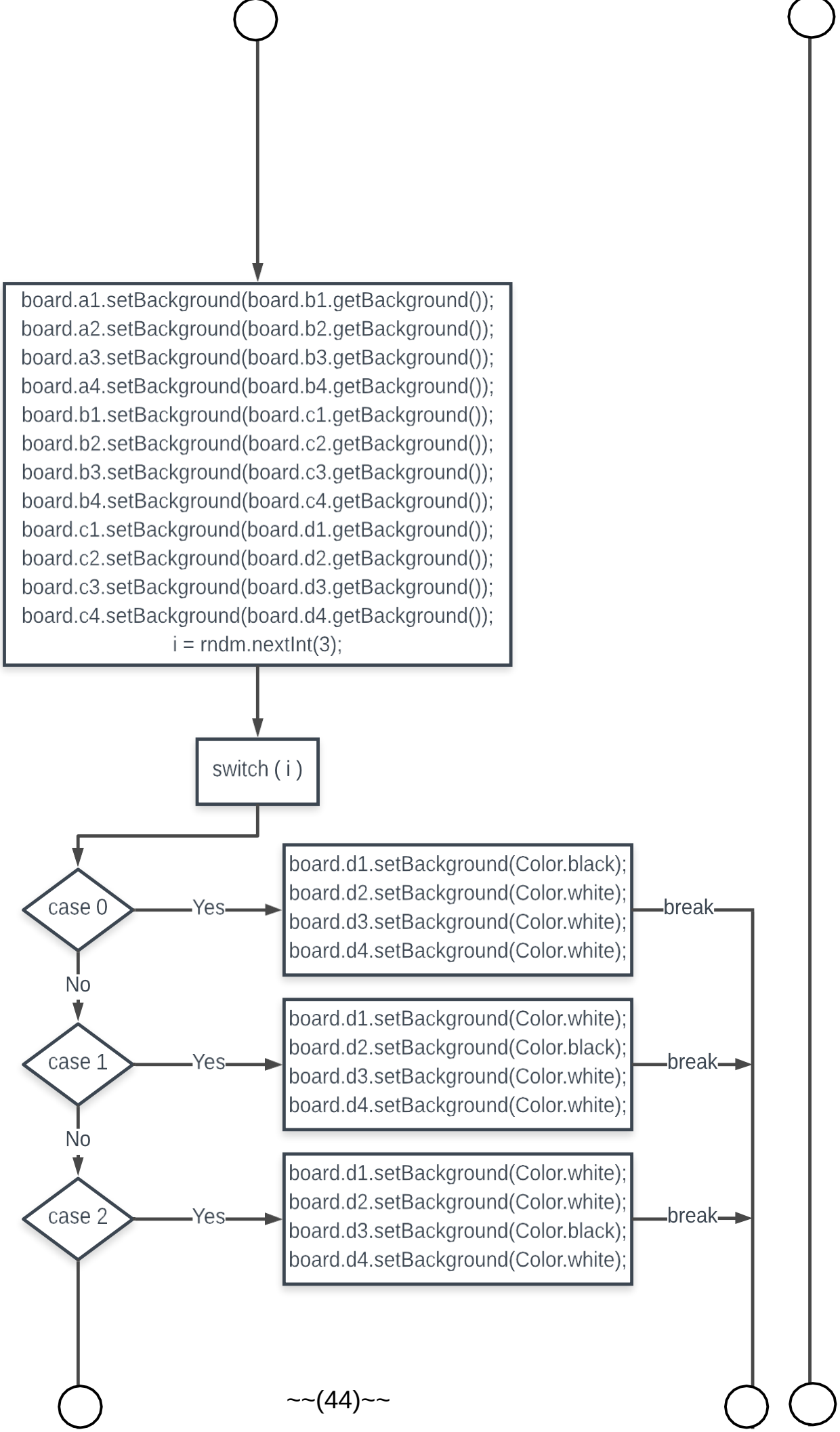

START-
Main_TA-
set_start()
int i;
i = rndm.nextInt(3);
switch (i)
case 0
Yes
board.d1.setBackground(Color.black);
board.d2.setBackground(Color.white);
board.d3.setBackground(Color.white);
board.d4.setBackground(Color.white);
break
No
case 1
Yes
board.d1.setBackground(Color.white);
board.d2.setBackground(Color.black);
board.d3.setBackground(Color.white);
board.d4.setBackground(Color.white);
break
No
case 2
Yes
board.d1.setBackground(Color.white);
board.d2.setBackground(Color.white);
board.d3.setBackground(Color.black);
board.d4.setBackground(Color.white);
break
No
case 3
Yes
board.d1.setBackground(Color.white);
board.d2.setBackground(Color.white);
board.d3.setBackground(Color.white);
board.d4.setBackground(Color.black);
break
No
i = rndm.nextInt(3);
switch(i)
case 0
Yes
board.c1.setBackground(Color.black);
board.c2.setBackground(Color.white);
board.c3.setBackground(Color.white);
board.c4.setBackground(Color.white);
break
No

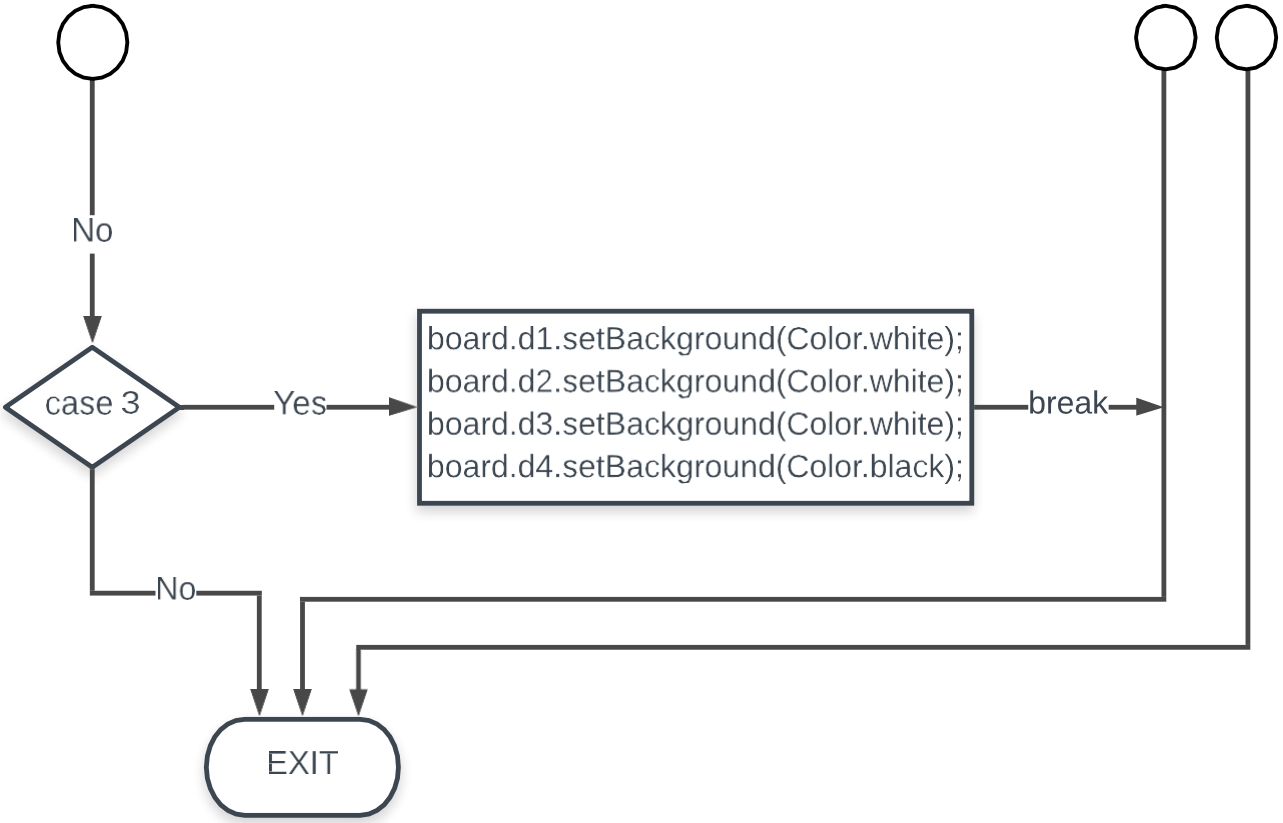

case 1
Yes
board.c1.setBackground(Color.black);
board.c2.setBackground(Color.white);
board.c3.setBackground(Color.white);
board.c4.setBackground(Color.white);
break
No
case 2
Yes
board.c1.setBackground(Color.black);
board.c2.setBackground(Color.white);
board.c3.setBackground(Color.white);
board.c4.setBackground(Color.white);
break
No
case 3
Yes
board.c1.setBackground(Color.black);
board.c2.setBackground(Color.white);
board.c3.setBackground(Color.white);
board.c4.setBackground(Color.white);
break
No
board.b1.setBackground(Color.white);
board.b2.setBackground(Color.white);
board.b3.setBackground(Color.white);
board.b4.setBackground(Color.white);
board.a1.setBackground(Color.white);
board.a2.setBackground(Color.white);
board.a3.setBackground(Color.white);
board.a4.setBackground(Color.white);
EXIT

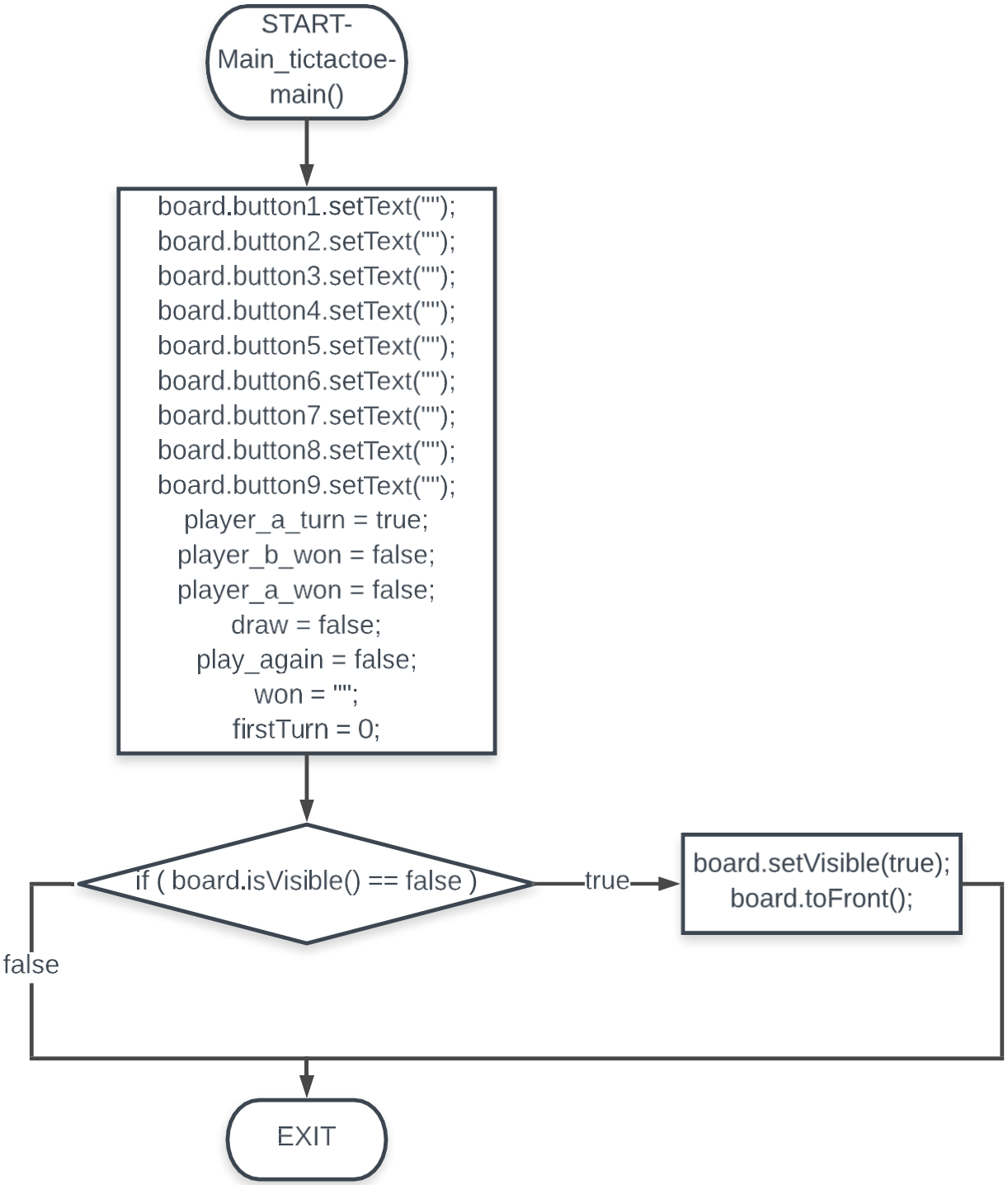

START-
Main_TA-
set_after()
if (
board.a1.getBackground
().equals(Color.black) ||
board.a2.getBackground
().equals(Color.black) ||
board.a3.getBackground
().equals(Color.black) ||
board.a4.getBackground
().equals(Color.black))
true
correct = false;
TA.timer.stop();
board.dispose();
Try block
2999
MILLISECONDS
Exception
occurs
true
false
catch
block
Logger.getLogger(Main_TA.
class.getName
()).log(Level.SEVERE, null, ex);
false
l3 =
(Long.toString(score));
Try block
=ileInputStream fis = new
FileInputStream("TA.txt");
ObjectInputStream ois = new
ObjectInputStream(fis);
highest = (long[])
ois.readObject();
ois.close();

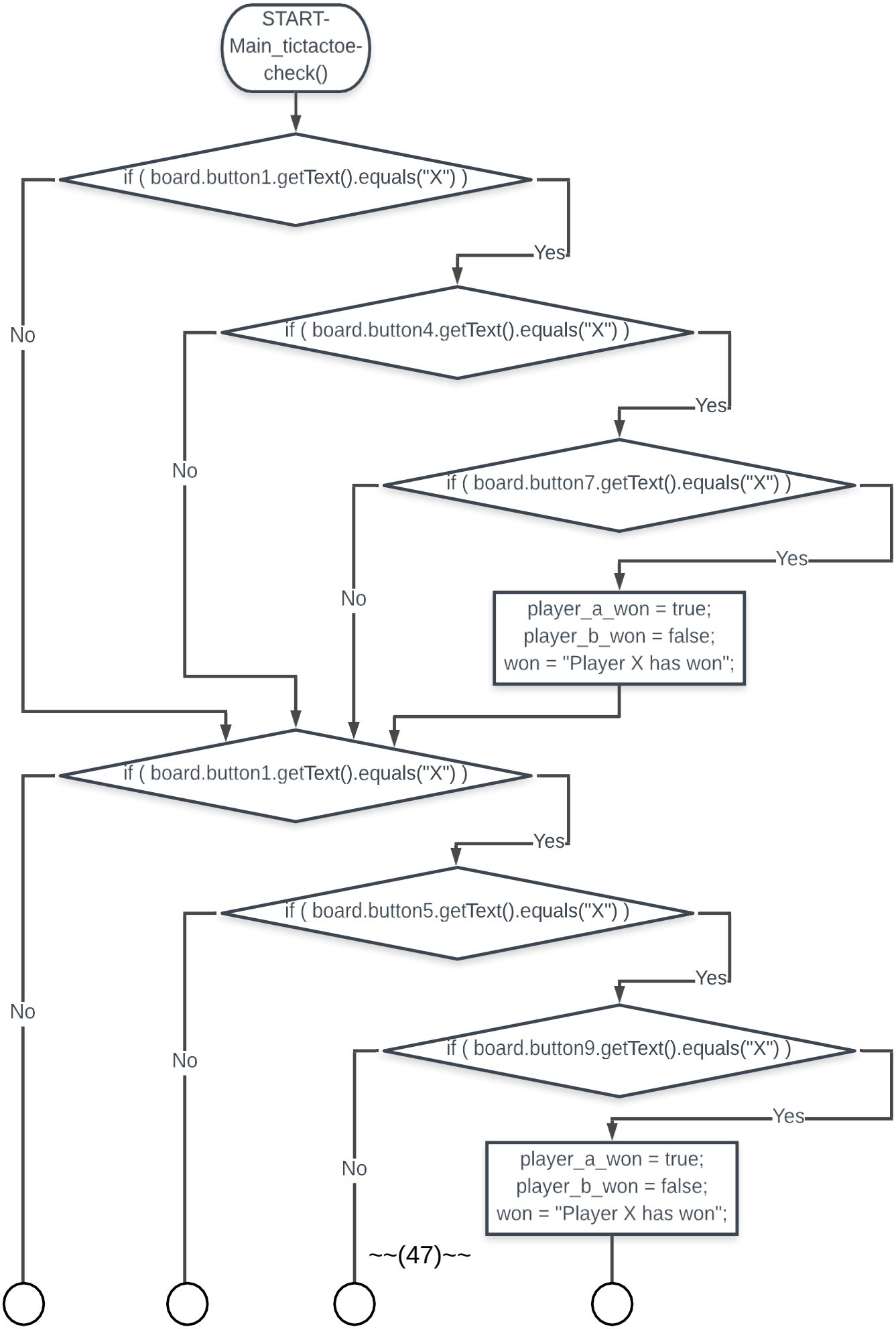

Exception occurs
true
catch block
false
if (highest[0] < score)
Yes
t1 = ("New highest scores.");
highest[0] = score;
Try block
FileOutputStream fos = new FileOutputStream("TA.txt");
ObjectOutputStream oos = new ObjectOutputStream(fos);
oos.writeObject(highest);
oos.close();
Exception occurs
true
catch block
false
No

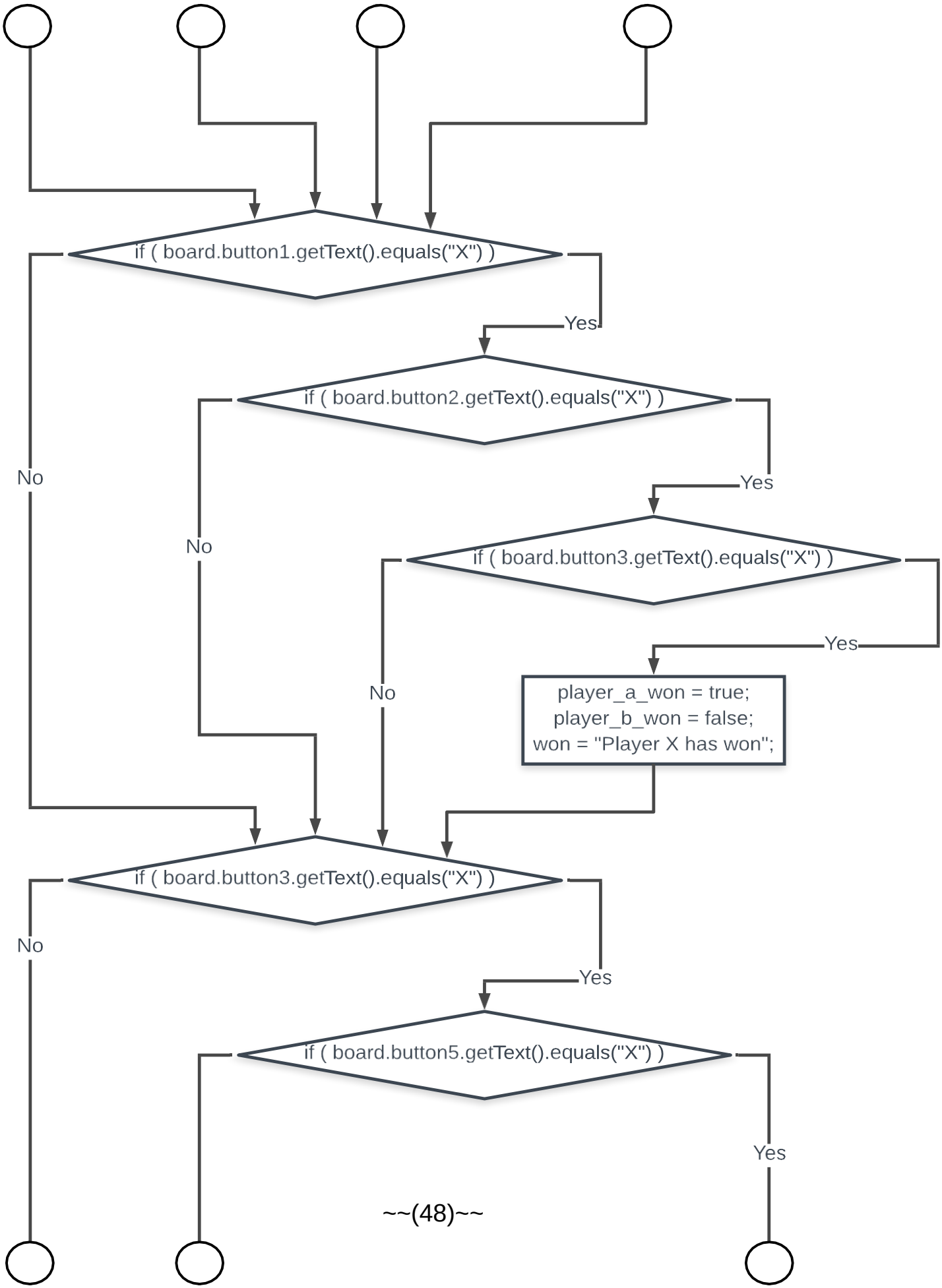

if (highest[0] > score)
Yes
t1 = ("Highest Scores are :- "); t2 = (Long.toString(highest[0]));
No
if (highest[0] == score)
Yes
t1 = ("Scores similar to Highest Scores");
No
Main.exit_a = true;
Main_TA-afterattack

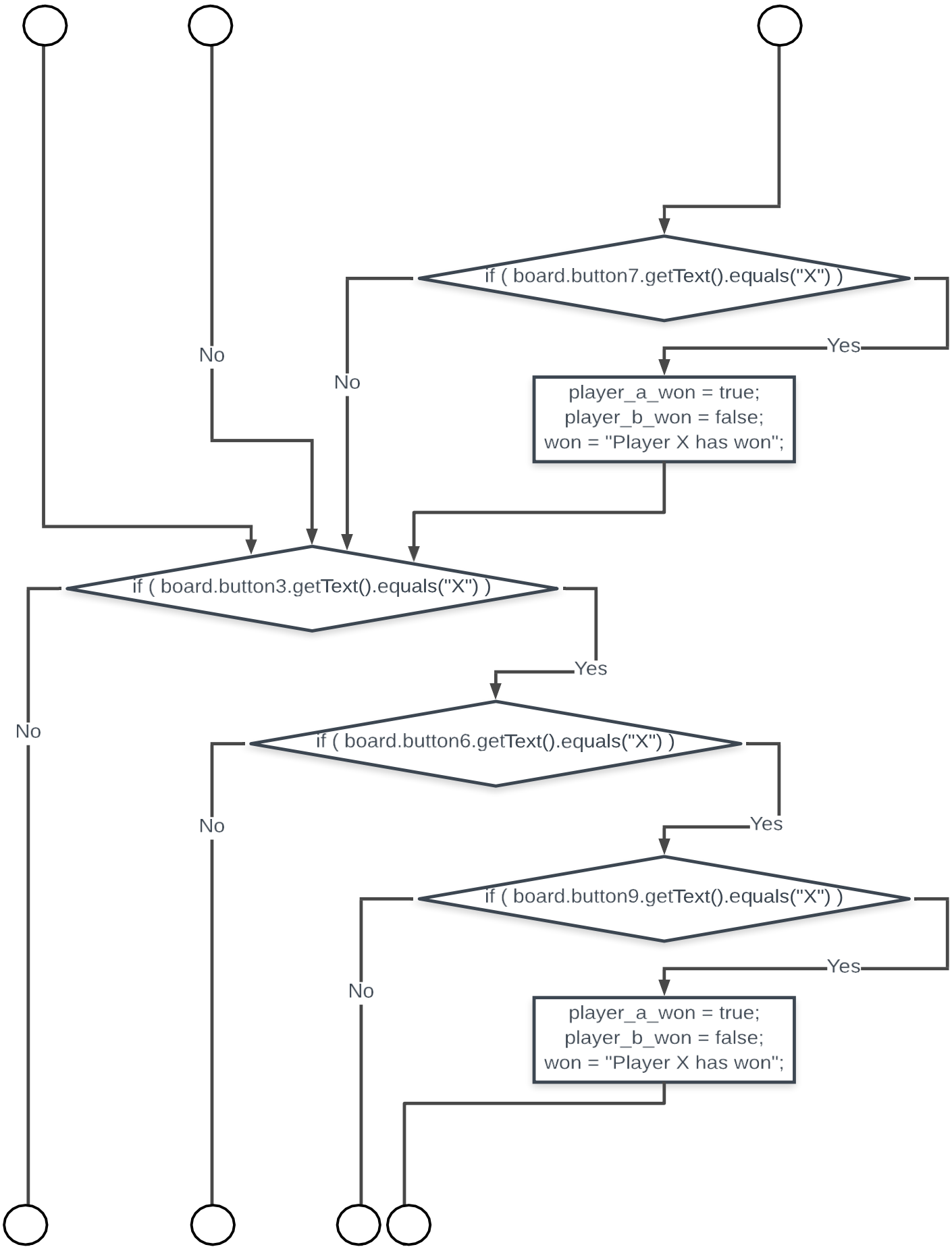

board.a1.setBackground(board.b1.getBackground());
board.a2.setBackground(board.b2.getBackground());
board.a3.setBackground(board.b3.getBackground());
board.a4.setBackground(board.b4.getBackground());
board.b1.setBackground(board.c1.getBackground());
board.b2.setBackground(board.c2.getBackground());
board.b3.setBackground(board.c3.getBackground());
board.b4.setBackground(board.c4.getBackground());
board.c1.setBackground(board.d1.getBackground());
board.c2.setBackground(board.d2.getBackground());
board.c3.setBackground(board.d3.getBackground());
board.c4.setBackground(board.d4.getBackground());
i = rndm.nextInt(3);
switch (i)
case 0
Yes
board.d1.setBackground(Color.black);
board.d2.setBackground(Color.white);
board.d3.setBackground(Color.white);
board.d4.setBackground(Color.white);
break
No
case 1
Yes
board.d1.setBackground(Color.white);
board.d2.setBackground(Color.black);
board.d3.setBackground(Color.white);
board.d4.setBackground(Color.white);
break
No
case 2
Yes
board.d1.setBackground(Color.white);
board.d2.setBackground(Color.white);
board.d3.setBackground(Color.black);
board.d4.setBackground(Color.white);
break

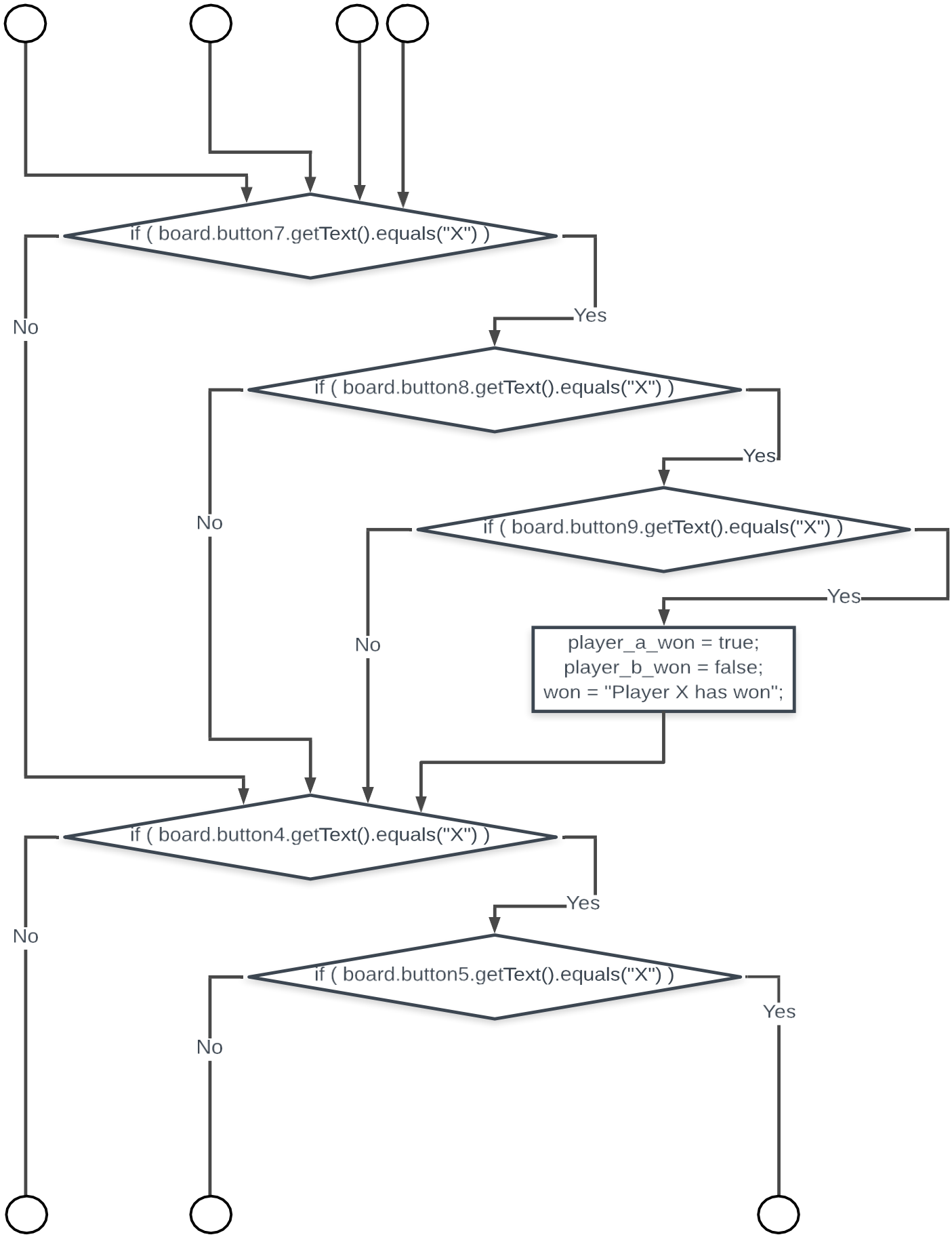

No
case 3
Yes
board.d1.setBackground(Color.white);
board.d2.setBackground(Color.white);
board.d3.setBackground(Color.white);
board.d4.setBackground(Color.black);
break
No
EXIT

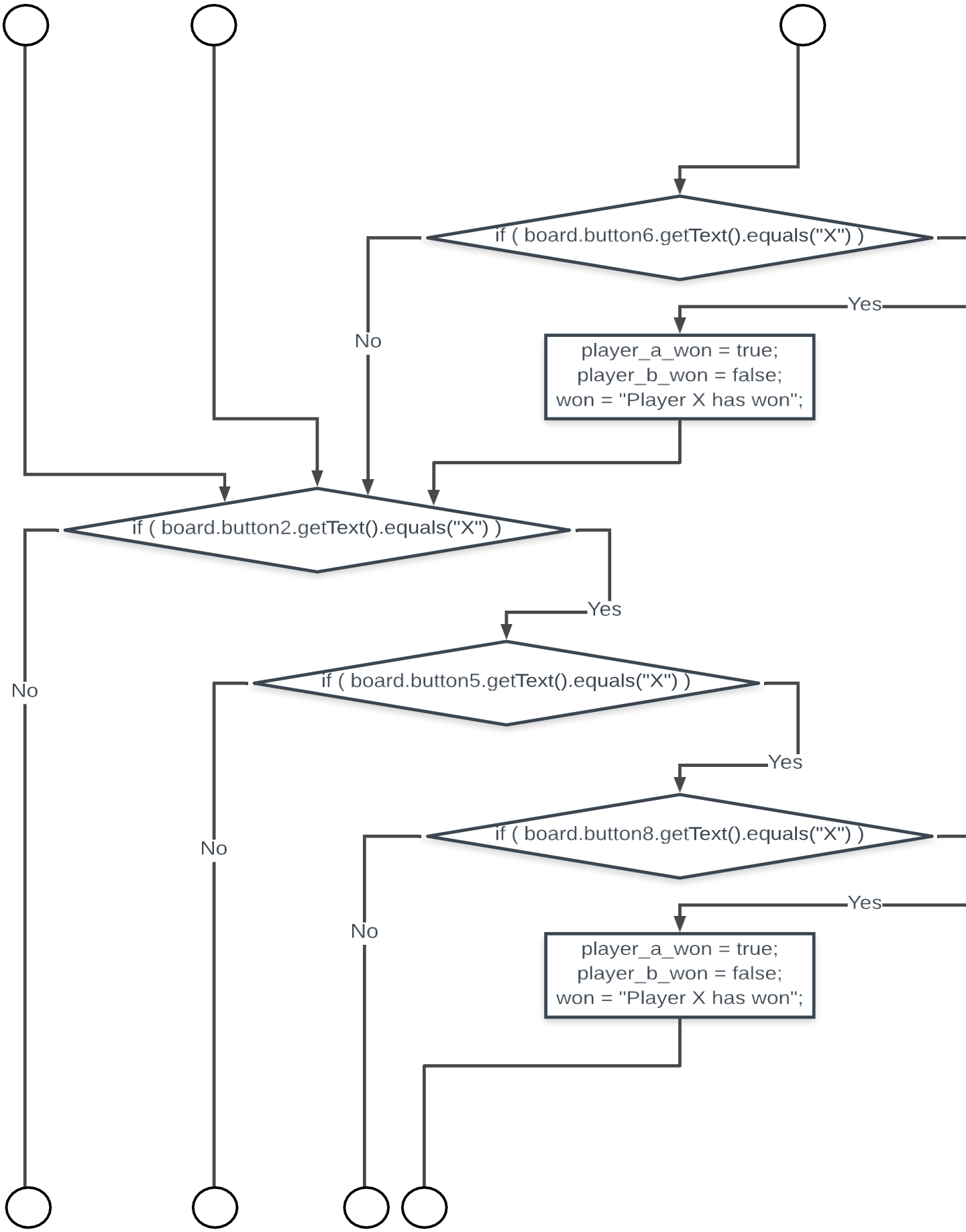

START-
Main_tictactoe-
main()
board.button1.setText("");
board.button2.setText("");
board.button3.setText("");
board.button4.setText("");
board.button5.setText("");
board.button6.setText("");
board.button7.setText("");
board.button8.setText("");
board.button9.setText("");
player_a_turn = true;
player_b_won = false;
player_a_won = false;
draw = false;
play_again = false;
won = "";
firstTurn = 0;
if (board.isVisible() == false)
true
board.setVisible(true);
board.toFront();
false
EXIT

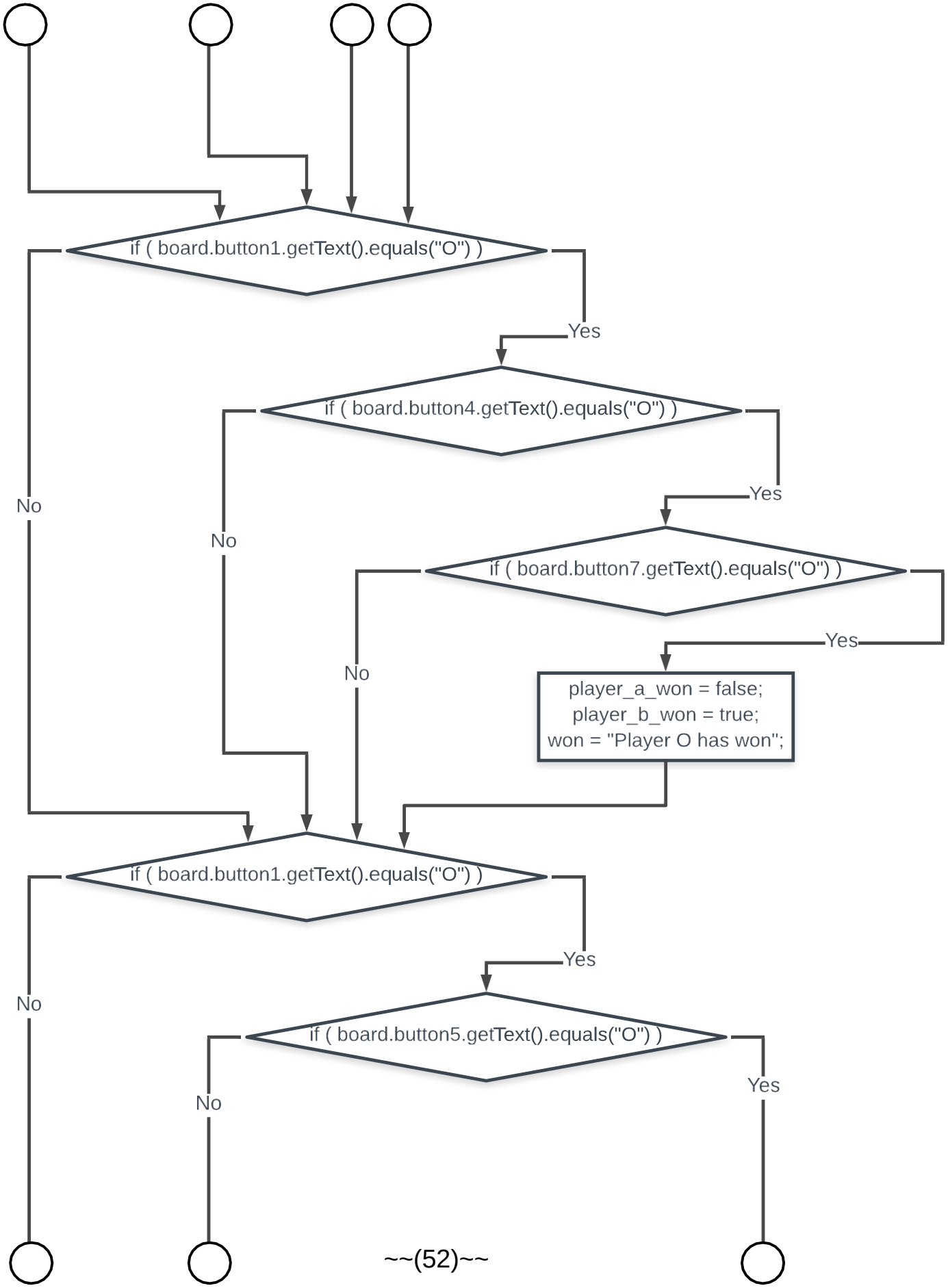

START-
Main_tictactoe-
check()
if (board.button1.getText().equals("X"))
Yes
No
if (board.button4.getText().equals("X"))
Yes
No
if (board.button7.getText().equals("X"))
Yes
No
player_a_won = true;
player_b_won = false;
won = "Player X has won";
if (board.button1.getText().equals("X"))
Yes
No
if (board.button5.getText().equals("X"))
Yes
No
if (board.button9.getText().equals("X"))
Yes
No
player_a_won = true;
player_b_won = false;
won = "Player X has won";

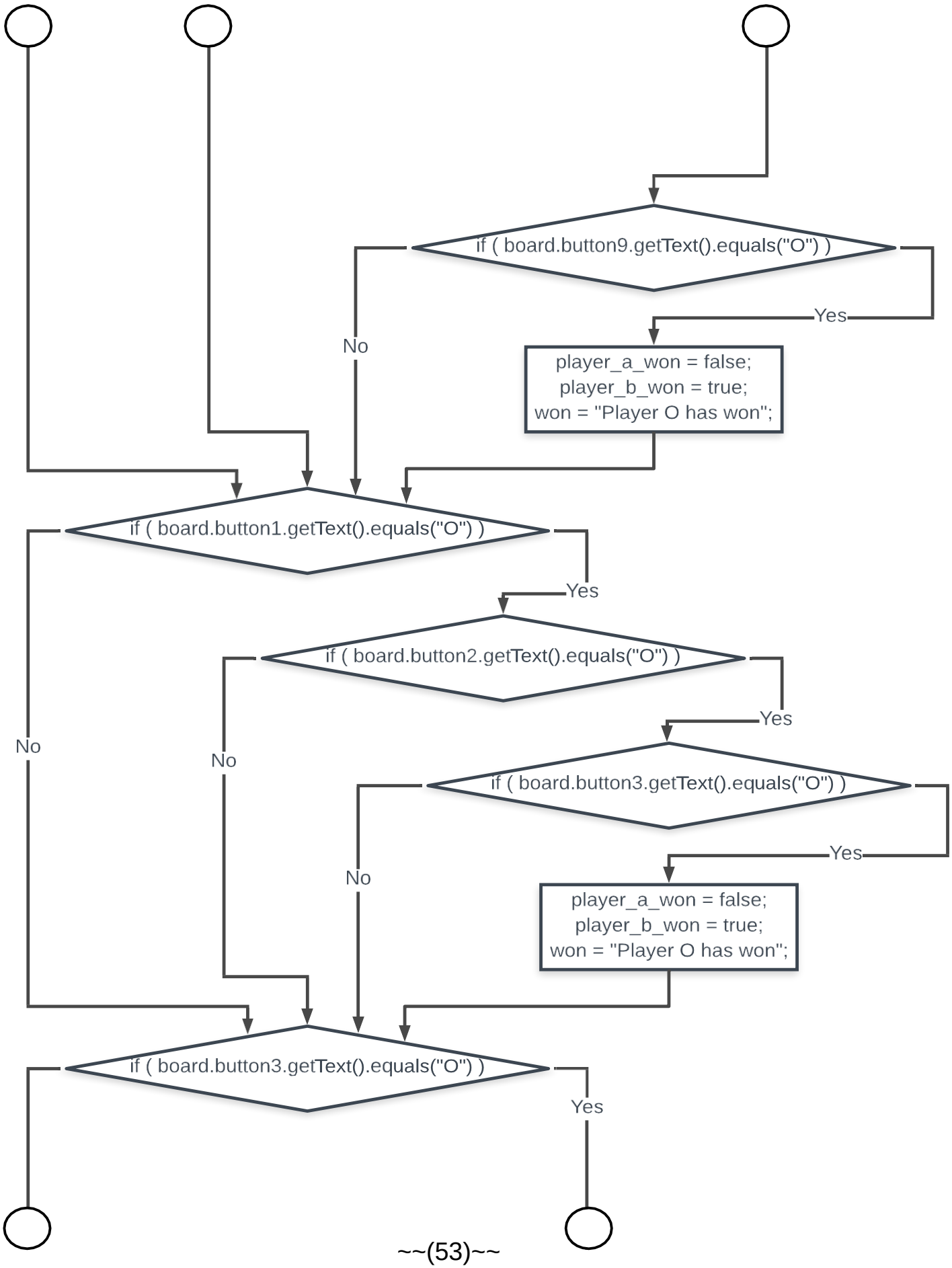

if (board.button1.getText().equals("X"))
Yes
No
if (board.button2.getText().equals("X"))
Yes
No
if (board.button3.getText().equals("X"))
Yes
No
player_a_won = true;
player_b_won = false;
won = "Player X has won";
if (board.button3.getText().equals("X"))
Yes
No
if (board.button5.getText().equals("X"))
Yes

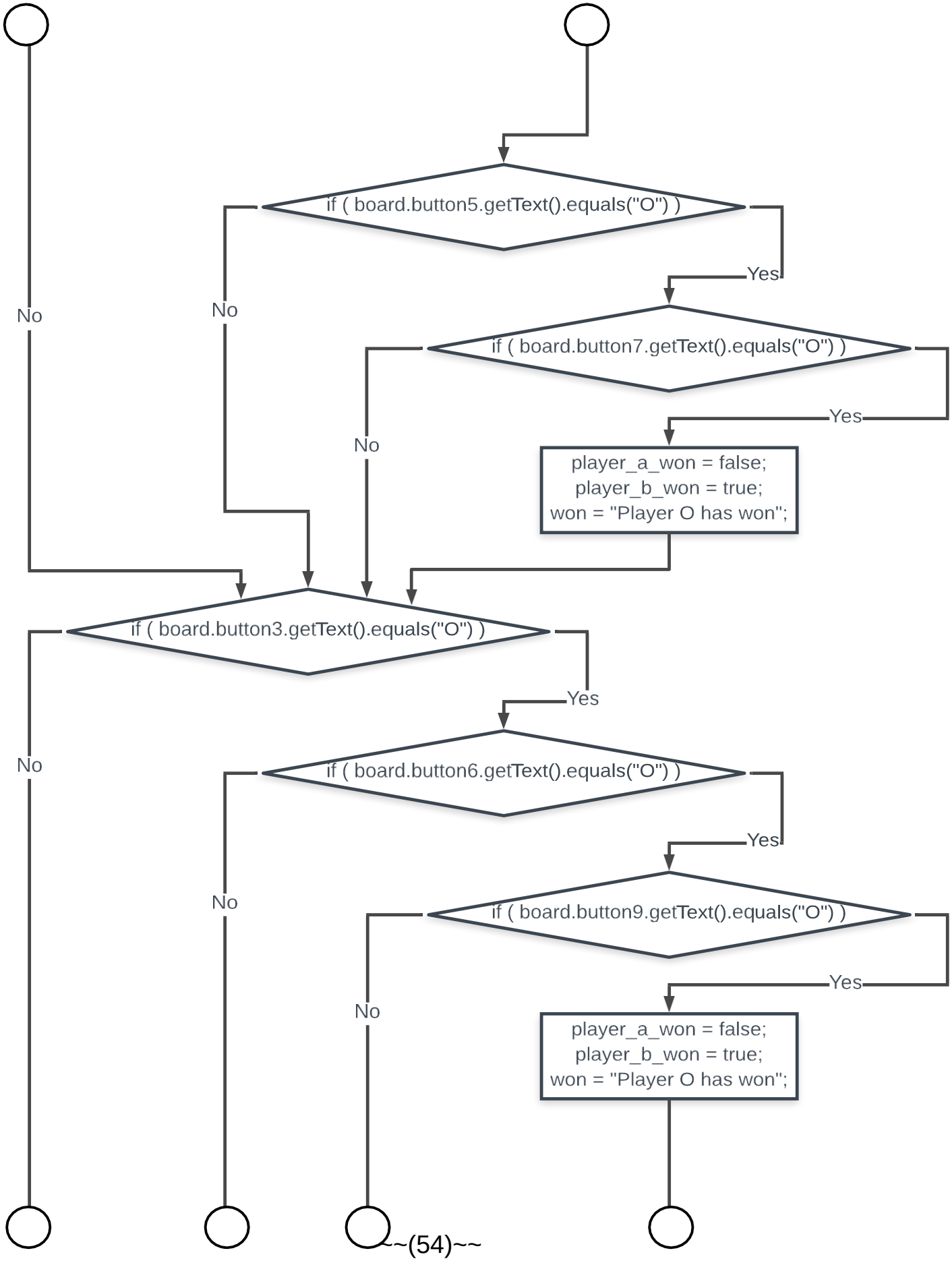

if (board.button7.getText().equals("X"))
Yes
No
No
player_a_won = true;
player_b_won = false;
won = "Player X has won";
if (board.button3.getText().equals("X"))
Yes
No
if (board.button6.getText().equals("X"))
No
Yes
if (board.button9.getText().equals("X"))
Yes
No
player_a_won = true;
player_b_won = false;
won = "Player X has won";

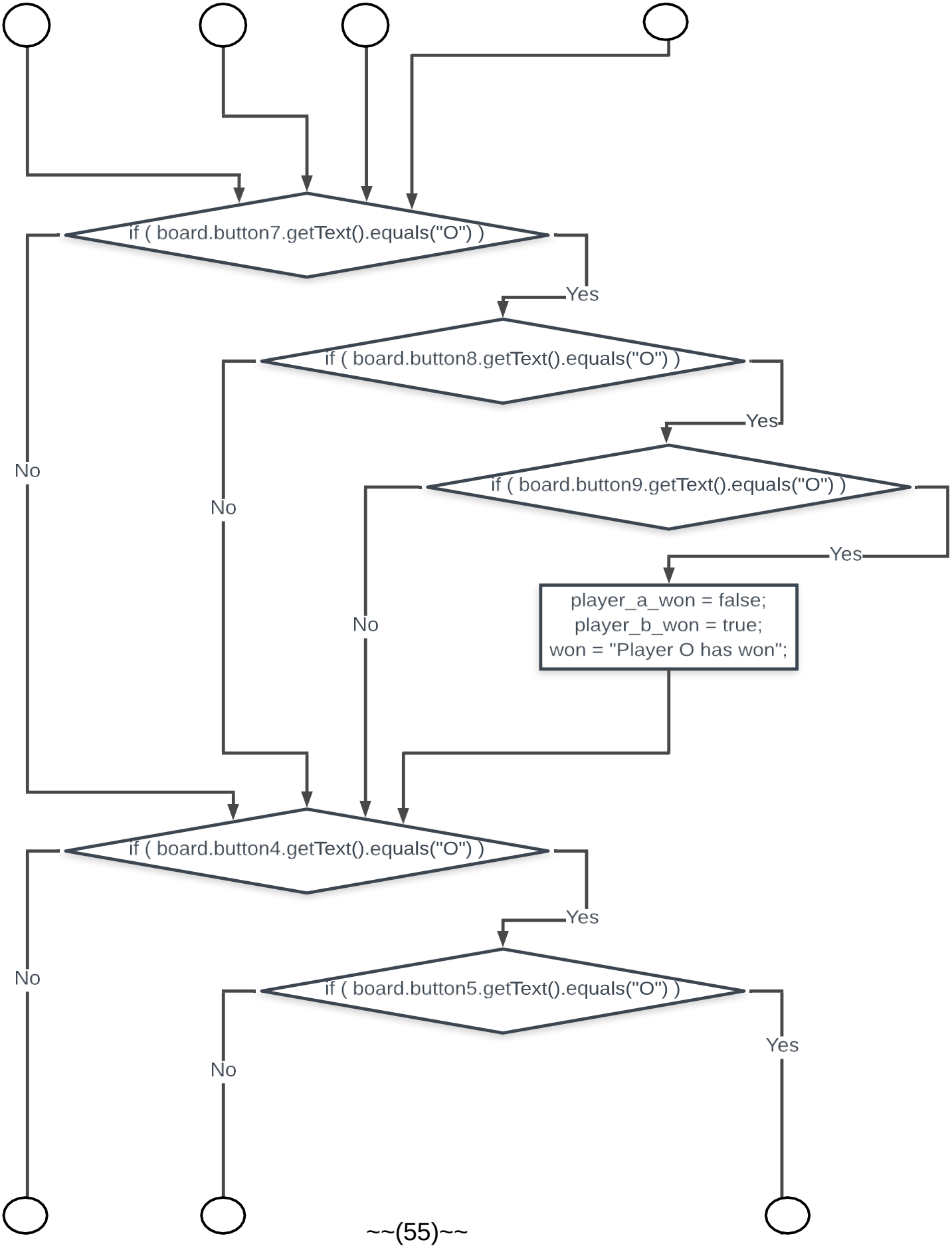

if (board.button7.getText().equals("X"))
Yes
No
if (board.button8.getText().equals("X"))
Yes
No
if (board.button9.getText().equals("X"))
Yes
No
player_a_won = true;
player_b_won = false;
won = "Player X has won";
if (board.button4.getText().equals("X"))
Yes
No
if (board.button5.getText().equals("X"))
Yes
No

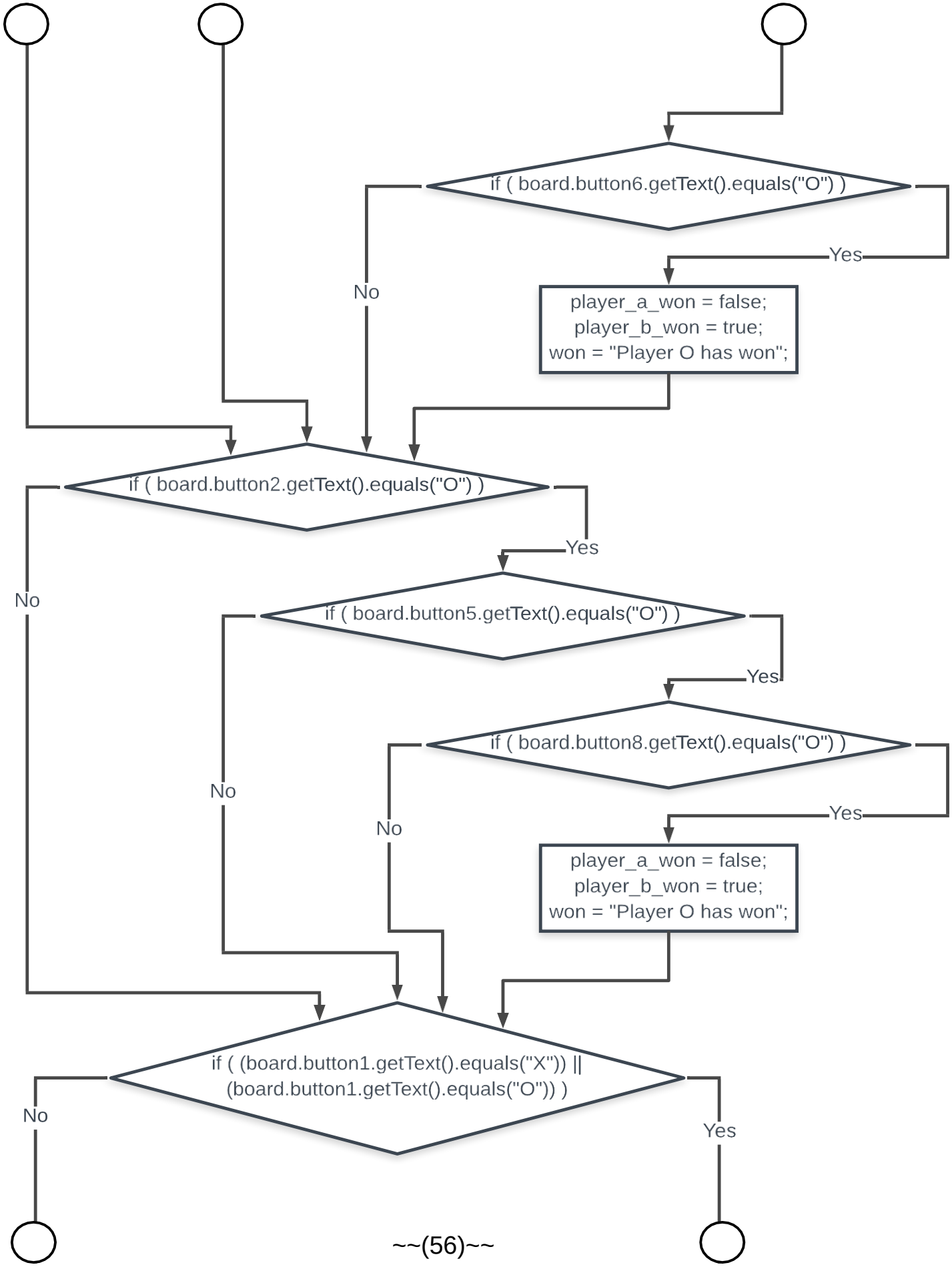

if (board.button6.getText().equals("X"))
Yes
No
player_a_won = true;
player_b_won = false;
won = "Player X has won";
if (board.button2.getText().equals("X"))
Yes
No
if (board.button5.getText().equals("X"))
Yes
No
if (board.button8.getText().equals("X"))
Yes
No
player_a_won = true;
player_b_won = false;
won = "Player X has won";

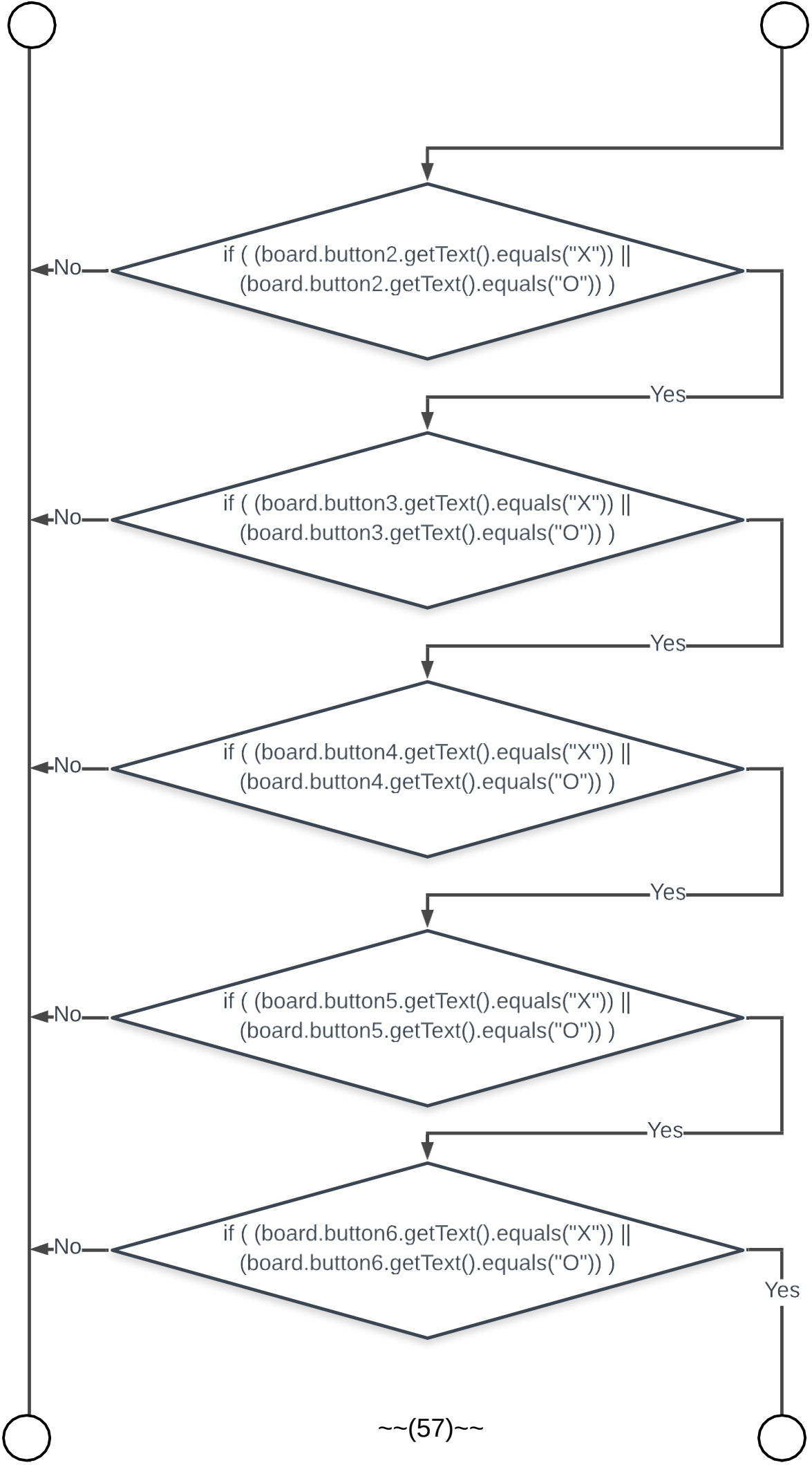

if (board.button1.getText().equals("O"))
Yes
No
if (board.button4.getText().equals("O"))
Yes
No
if (board.button7.getText().equals("O"))
Yes
No
player_a_won = false;
player_b_won = true;
won = "Player O has won";
if (board.button1.getText().equals("O"))
Yes
No
if (board.button5.getText().equals("O"))
No
Yes

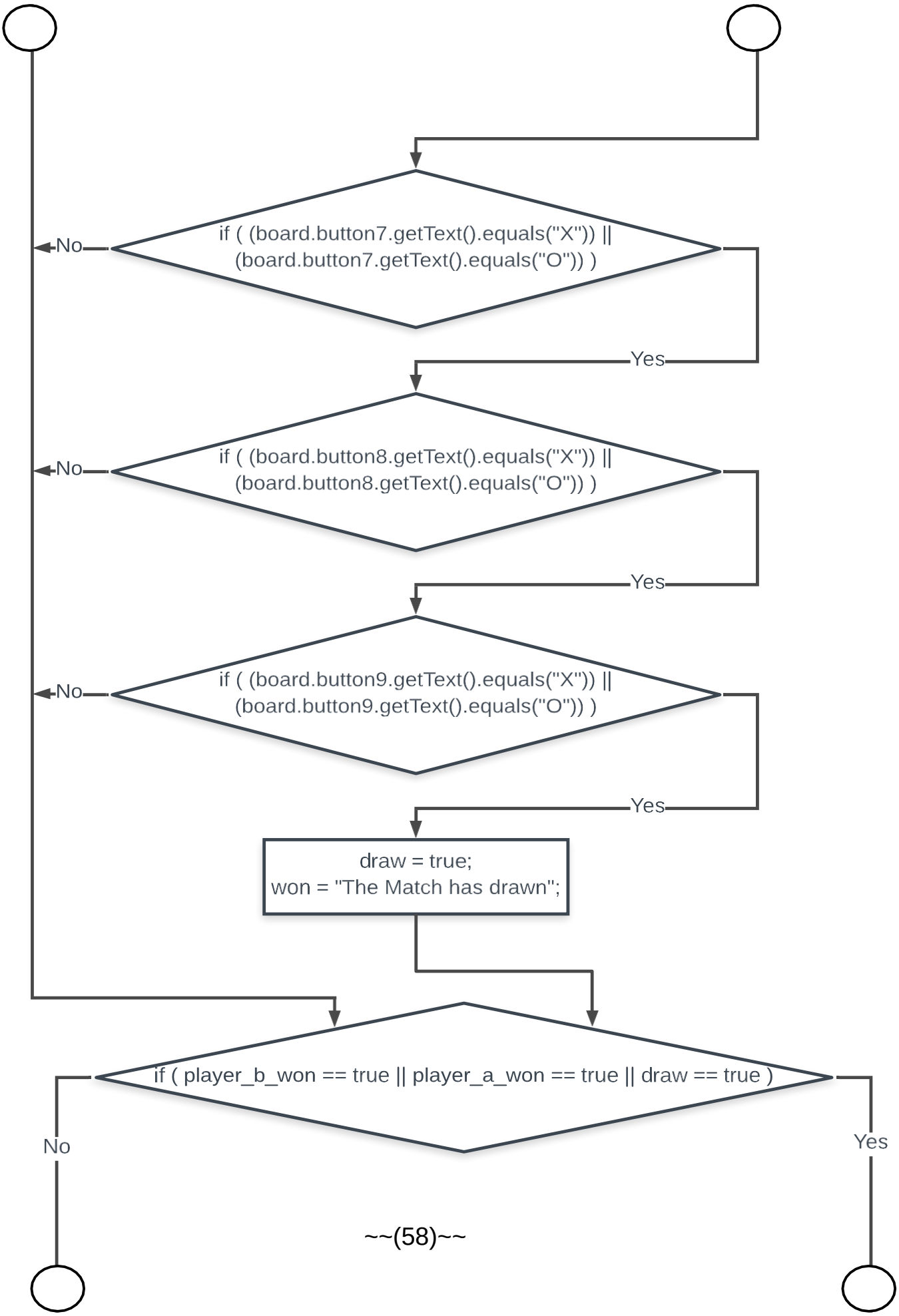

if (board.button9.getText().equals("O"))
Yes
No
player_a_won = false;
player_b_won = true;
won = "Player O has won";
if (board.button1.getText().equals("O"))
Yes
No
if (board.button2.getText().equals("O"))
Yes
No
if (board.button3.getText().equals("O"))
Yes
No
player_a_won = false;
player_b_won = true;
won = "Player O has won";
if (board.button3.getText().equals("O"))
Yes

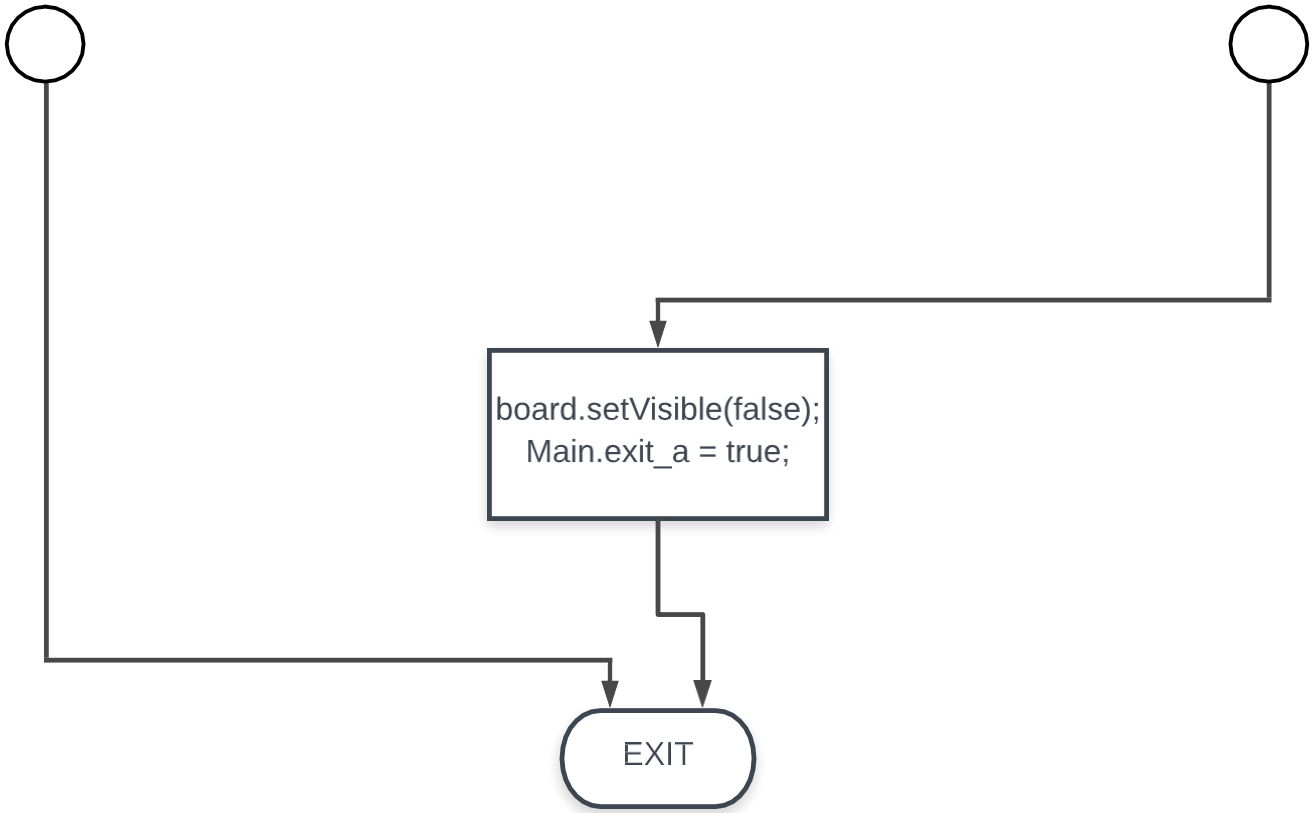

if (board.button5.getText().equals("O"))
Yes
No
No
if (board.button7.getText().equals("O"))
Yes
No
player_a_won = false;
player_b_won = true;
won = "Player O has won";
if (board.button3.getText().equals("O"))
Yes
No
if (board.button6.getText().equals("O"))
Yes
No
if (board.button9.getText().equals("O"))
Yes
No
player_a_won = false;
player_b_won = true;
won = "Player O has won";


if (board.button7.getText().equals("O"))
Yes
No
if (board.button8.getText().equals("O"))
Yes
No
if (board.button9.getText().equals("O"))
Yes
No
player_a_won = false;
player_b_won = true;
won = "Player O has won";
if (board.button4.getText().equals("O"))
Yes
No
if (board.button5.getText().equals("O"))
Yes
No


if (board.button6.getText().equals("O"))
Yes
No
player_a_won = false;
player_b_won = true;
won = "Player O has won";
if (board.button2.getText().equals("O"))
Yes
No
if (board.button5.getText().equals("O"))
Yes
No
if (board.button8.getText().equals("O"))
Yes
No
player_a_won = false;
player_b_won = true;
won = "Player O has won";
if ((board.button1.getText().equals("X")) ||
(board.button1.getText().equals("O")))
No
Yes


if ((board.button2.getText().equals("X")) ||
(board.button2.getText().equals("O")))
No
Yes
if ((board.button3.getText().equals("X")) ||
(board.button3.getText().equals("O")))
No
Yes
if ((board.button4.getText().equals("X")) ||
(board.button4.getText().equals("O")))
No
Yes
if ((board.button5.getText().equals("X")) ||
(board.button5.getText().equals("O")))
No
Yes
if ((board.button6.getText().equals("X")) ||
(board.button6.getText().equals("O")))
No
Yes


if ((board.button7.getText().equals("X")) ||
(board.button7.getText().equals("O")))
No
Yes
if ((board.button8.getText().equals("X")) ||
(board.button8.getText().equals("O")))
No
Yes
if ((board.button9.getText().equals("X")) ||
(board.button9.getText().equals("O")))
No
Yes
draw = true;
won = "The Match has drawn";
if (player_b_won == true || player_a_won == true || draw == true)
No
Yes


board.setVisible(false);
Main.exit_a = true;
EXIT

Source Code

```
import java.net.URISyntaxException;
import java.util.concurrent.TimeUnit;

public class Main {

    public static boolean exit_a = false;

    private static TimeUnit Unit = TimeUnit.MILLISECONDS;

    public static byte i = -1;

    private static void start() throws InterruptedException {
        TimeUnit Unit = TimeUnit.MILLISECONDS;
        Start board = new Start();
        if ( board.isVisible() == false ) {
            board.setVisible(true);
            board.toFront();
        }
        board.l1.setText("   B");
        Unit.sleep(1000);
        board.l2.setText("     &");
        Unit.sleep(1000);
        board.l3.setText("       B");
        Unit.sleep(1000);
        board.l4.setText("Game Library");
        Unit.sleep(2000);
```

```
    board.dispose();
}

public static void exit() throws InterruptedException {
    TimeUnit Unit = TimeUnit.MILLISECONDS;
    Exit exit = new Exit();
    if ( exit.isVisible() == false ) {
        exit.setVisible(true);
        exit.toFront();
    }
    Unit.sleep(3500);
    exit.dispose();
    System.exit(0);
}

public static void play() throws InterruptedException {
    SpacePiano board1 = new SpacePiano();
    Tictactoe board2 = new Tictactoe();
    TA board3 = new TA();
    switch ( i ) {
        case 0:
            exit();
            Unit.sleep(2000);
            System.exit(0);
            break;
        case 1:
```

```
        Main_SpacePiano.main();
        while ( exit_a == false ) {
            if ( exit_a == true ) {
                break;
            }
        }
        board1.dispose();
        SP_Scores b = new SP_Scores();
        if ( b.isVisible() == false ) {
            b.setVisible(true);
            b.toFront();
        }
        Unit.sleep(4500);
        b.dispose();
        exit_a = true;
        break;
    case 2:
        Main_tictactoe.main();
        while ( exit_a == false ) {
            if ( exit_a == true ) {
                break;
            }
        }
        board2.dispose();
        TTT_Scores t = new TTT_Scores();
        if ( t.isVisible() == false ) {
```

```
            t.setVisible(true);
            t.toFront();
        }
        Unit.sleep(4200);
        t.dispose();
        exit_a = true;
        break;
    case 3:
        Main_TA.main();
        while ( exit_a == false ) {
            if ( exit_a == true ) {
                break;
            }
        }
        board3.dispose();
        TA_Scores a = new TA_Scores();
        if ( a.isVisible() == false ) {
            a.setVisible(true);
            a.toFront();
        }
        Unit.sleep(4500);
        a.dispose();
        exit_a = true;
        break;
}
Unit.sleep(270);
```

```
        i = -1;
        exit_a = false;
    }

    public static void main(String[] args) throws InterruptedException,
URISyntaxException {
        start();
        SplashScreen sc1 = new SplashScreen("B.png");
        Unit.sleep(1800);
        sc1.dispose();
        SplashScreen sc2 = new SplashScreen("C.png");
        Unit.sleep(1800);
        sc2.dispose();
        SplashScreen sc3 = new SplashScreen("A.png");
        Unit.sleep(1800);
        sc3.dispose();
        ProductKey pk = new ProductKey();
        while ( ProductKey.cont == 0 ) {
            if ( ProductKey.cont == 1 ) {
                break;
            }
        }
        pk.dispose();
        byte times = 0;
        while ( times < 2 ) {
            Menu m = new Menu();
```

```
            if ( m.isVisible() == false ) {
            m.setVisible(true);
            m.toFront();
            }
            while ( i < 0 || i >= 4 ) {
            }
            m.dispose();
            play();
            times++;
        }
        exit();
        Unit.sleep(2000);
        System.exit(0);
    }
}
```

```
import java.awt.Container;
import javax.swing.*;

public class Start extends JFrame {

    Start() {
        initComponents1();
    }

    void initComponents1() {
        l1 = new JLabel();
        l2 = new JLabel();
        l3 = new JLabel();
        l4 = new JLabel();


setDefaultCloseOperation(javax.swing.WindowConstants.DO_NOTHING_
ON_CLOSE);
        setResizable(false);
        setUndecorated(true);
        setAlwaysOnTop(true);

        setTitle("");
        Container contentPane = getContentPane();
```

```
        l1.setText("");
        l2.setText("");
        l3.setText("");
        l4.setText("");

        GroupLayout contentPaneLayout = new GroupLayout(contentPane);
        contentPane.setLayout(contentPaneLayout);
        contentPaneLayout.setHorizontalGroup(
                contentPaneLayout.createParallelGroup()
                        .addGroup(contentPaneLayout.createSequentialGroup()
                                .addContainerGap()
                                .addGroup(contentPaneLayout.createParallelGroup()

.addGroup(contentPaneLayout.createSequentialGroup()
                                        .addGap(79)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)
                                        .addComponent(l1,
GroupLayout.PREFERRED_SIZE, 130, GroupLayout.PREFERRED_SIZE)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED))

.addGroup(contentPaneLayout.createSequentialGroup()
                                        .addGap(79)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)
                                        .addComponent(l2,
GroupLayout.PREFERRED_SIZE, 130, GroupLayout.PREFERRED_SIZE)
```

```
.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED))

.addGroup(contentPaneLayout.createSequentialGroup()
                                        .addGap(79)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)
                                        .addComponent(l3,
GroupLayout.PREFERRED_SIZE, 130, GroupLayout.PREFERRED_SIZE)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED))

.addGroup(contentPaneLayout.createSequentialGroup()
                                        .addGap(69)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)
                                        .addComponent(l4,
GroupLayout.PREFERRED_SIZE, 130, GroupLayout.PREFERRED_SIZE)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)))
                                .addContainerGap(11, Short.MAX_VALUE))
        );
        contentPaneLayout.setVerticalGroup(
                contentPaneLayout.createParallelGroup()
                        .addGroup(contentPaneLayout.createSequentialGroup()
                                .addContainerGap()

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)
```

```
.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)
                                .addComponent(l1,
GroupLayout.PREFERRED_SIZE, 30, GroupLayout.PREFERRED_SIZE))

.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)
                                .addComponent(l2,
GroupLayout.PREFERRED_SIZE, 30, GroupLayout.PREFERRED_SIZE))

.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)
                                .addComponent(l3,
GroupLayout.PREFERRED_SIZE, 30, GroupLayout.PREFERRED_SIZE))

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)

.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)
                                .addComponent(l4,
GroupLayout.PREFERRED_SIZE, 30, GroupLayout.PREFERRED_SIZE))
                        .addContainerGap(15, Short.MAX_VALUE))
        );
        pack();
        setLocationRelativeTo(getOwner());
    }

    JButton button1, button2, button3;
```

```
    JLabel l1, l2, l3, l4;

}
```

```
import java.awt.Container;
import javax.swing.*;

public class SplashScreen extends JFrame {

    static String image;

    public SplashScreen(String i)  {
        image = i;
        initComponents();
        if ( isVisible() == false ) {
            setVisible(true);
        }
    }

    public void initComponents() {
        ImageIcon i = new ImageIcon(image);
        l1 = new JLabel(i);

setDefaultCloseOperation(javax.swing.WindowConstants.DO_NOTHING_
ON_CLOSE);
        setUndecorated(true);
        setAlwaysOnTop(true);
```

```
        setResizable(false);


        setTitle("");
        Container contentPane = getContentPane();



        GroupLayout contentPaneLayout = new GroupLayout(contentPane);
        contentPane.setLayout(contentPaneLayout);
        contentPaneLayout.setHorizontalGroup(
                contentPaneLayout.createParallelGroup()
                        .addGroup(contentPaneLayout.createSequentialGroup()
                                .addContainerGap()
                                .addGroup(contentPaneLayout.createParallelGroup()

.addGroup(contentPaneLayout.createSequentialGroup()
                                        .addComponent(l1,
GroupLayout.PREFERRED_SIZE, 350,
GroupLayout.PREFERRED_SIZE)))
                                .addContainerGap(9, Short.MAX_VALUE))
        );
        contentPaneLayout.setVerticalGroup(
                contentPaneLayout.createParallelGroup()
                        .addGroup(contentPaneLayout.createSequentialGroup()
                                .addContainerGap()

.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)
```

```
                    .addComponent(l1,
GroupLayout.PREFERRED_SIZE, 200, GroupLayout.PREFERRED_SIZE))
                    .addContainerGap(9, Short.MAX_VALUE))
        );
        pack();
        setLocationRelativeTo(getOwner());

    }

    public JLabel l1;
}
```

```
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.IOException;
import java.net.URI;
import java.net.URISyntaxException;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.*;

public class ProductKey extends JFrame {

    ProductKey() throws URISyntaxException {
        initComponents();
        setVisible(true);
    }

    public void b2actionPerformed(ActionEvent e) throws
URISyntaxException {
        final URI uri = new URI("https://www.theunpublished.co.in");
        open(uri);
    }

    public void b1actionPerformed(ActionEvent e) {
        dispose();
```

```
        cont=1;
    }

    void initComponents() throws URISyntaxException {
        final URI uri = new URI("https://www.theunpublished.co.in");

        l1 = new JLabel();
        l2 = new JLabel();
        b1 = new JButton();
        b2 = new JButton();

        l1.setText("You are in Trial Version");
        l2.setText("You can run only any two games one time each, or one
game two times in one run.");
        b1.setText("Continue");

        b2.setText("<HTML>Buy <FONT
color=\"#00099\"><U>now</U></FONT>.</HTML>");
        b2.setHorizontalAlignment(SwingConstants.LEFT);
        b2.setBorderPainted(false);
        b2.setOpaque(false);
        b2.setBackground(Color.WHITE);
        b2.setToolTipText(uri.toString());

        b2.addActionListener(new ActionListener() {
            @Override
```

```
            public void actionPerformed(ActionEvent e) {
                try {
                    b2actionPerformed(e);
                } catch ( URISyntaxException ex ) {

Logger.getLogger(ProductKey.class.getName()).log(Level.SEVERE, null,
ex);
                }
            }
        });
        b1.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                b1actionPerformed(e);
            }
        });



setDefaultCloseOperation(javax.swing.WindowConstants.DO_NOTHING_
ON_CLOSE);
        setCursor(Cursor.HAND_CURSOR);
        setResizable(false);
        setAlwaysOnTop(true);


        setTitle("Buy Full Version");
        Container contentPane = getContentPane();
```

```
GroupLayout contentPaneLayout = new GroupLayout(contentPane);
contentPane.setLayout(contentPaneLayout);
contentPaneLayout.setHorizontalGroup(
    contentPaneLayout.createParallelGroup()
        .addGroup(contentPaneLayout.createSequentialGroup()
            .addContainerGap()
            .addGroup(contentPaneLayout.createParallelGroup()

.addGroup(contentPaneLayout.createSequentialGroup()
                .addComponent(l1,
GroupLayout.PREFERRED_SIZE, 268, GroupLayout.PREFERRED_SIZE)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED))

.addGroup(contentPaneLayout.createSequentialGroup()
                .addComponent(l2,
GroupLayout.PREFERRED_SIZE, 500, GroupLayout.PREFERRED_SIZE)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED))

.addGroup(contentPaneLayout.createSequentialGroup()
                .addComponent(b1,
GroupLayout.PREFERRED_SIZE, 100, GroupLayout.PREFERRED_SIZE)
                .addGap(13)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(b2,
GroupLayout.PREFERRED_SIZE, 100,
GroupLayout.PREFERRED_SIZE)))
            .addContainerGap(11, Short.MAX_VALUE))
```

```
        );
        contentPaneLayout.setVerticalGroup(
                contentPaneLayout.createParallelGroup()
                        .addGroup(contentPaneLayout.createSequentialGroup()
                                .addContainerGap()

.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)
                                        .addComponent(l1,
GroupLayout.PREFERRED_SIZE, 55, GroupLayout.PREFERRED_SIZE))

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)

.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)
                                        .addComponent(l2,
GroupLayout.PREFERRED_SIZE, 55, GroupLayout.PREFERRED_SIZE))

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)

.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)
                                        .addComponent(b1,
GroupLayout.PREFERRED_SIZE, 55, GroupLayout.PREFERRED_SIZE)
                                        .addGap(13)
                                        .addComponent(b2,
GroupLayout.PREFERRED_SIZE, 55, GroupLayout.PREFERRED_SIZE))
                                .addContainerGap(15, Short.MAX_VALUE))
        );
        pack();
```

```
        setLocationRelativeTo(getOwner());
    }

    private static void open(URI uri) {
        if ( Desktop.isDesktopSupported() ) {
            try {
                Desktop.getDesktop().browse(uri);
            } catch ( IOException e ) {
            }
        }
    }

    JLabel l1, l2;

    JButton b1, b2;

    public static byte cont=0;
}
```

```
import java.awt.Container;
import java.awt.Cursor;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;

public class Menu extends JFrame {

    Menu() {
        initComponents();
    }

    void spActionPerformed(ActionEvent e) {
        Main.i = 1;
    }

    void tttActionPerformed(ActionEvent e) {
        Main.i = 2;
    }

    void taActionPerformed(ActionEvent e) {
        Main.i = 3;
    }
```

```
    void exActionPerformed(ActionEvent e) {
        Main.i = 0;
    }

    void initComponents() {
        sp = new JButton();
        ttt = new JButton();
        ta = new JButton();
        ex = new JButton();
        l1 = new JLabel();
        l2 = new JLabel();
        l3 = new JLabel();


setDefaultCloseOperation(javax.swing.WindowConstants.DO_NOTHING_
ON_CLOSE);
        setCursor(Cursor.HAND_CURSOR);
        setResizable(false);
        setAlwaysOnTop(true);

        setTitle("B&B Game Library");
        Container contentPane = getContentPane();

        l1.setText("Welcome to the Game Library of Bhav !");
        l2.setText("Which Game would you like to play ?");
        l3.setText("B & B Game Library");
```

```
sp.setText("Space Piano");
sp.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        spActionPerformed(e);
    }
});
ttt.setText("Tic Tac Toe");
ttt.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        tttActionPerformed(e);
    }
});
ta.setText("Tiny Army");
ta.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        taActionPerformed(e);
    }
});
ex.setText("Exit");
ex.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
```

```
            exActionPerformed(e);
        }
    });

    GroupLayout contentPaneLayout = new GroupLayout(contentPane);
    contentPane.setLayout(contentPaneLayout);
    contentPaneLayout.setHorizontalGroup(
            contentPaneLayout.createParallelGroup()
                    .addGroup(contentPaneLayout.createSequentialGroup()
                            .addContainerGap()
                            .addGroup(contentPaneLayout.createParallelGroup()

.addGroup(contentPaneLayout.createSequentialGroup()
                                    .addGap(31)
                                    .addComponent(l1,
GroupLayout.PREFERRED_SIZE, 250, GroupLayout.PREFERRED_SIZE)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)
                                    .addGap(69)
                                    .addComponent(ex,
GroupLayout.PREFERRED_SIZE, 89, GroupLayout.PREFERRED_SIZE))
                                .addGap(39)

.addGroup(contentPaneLayout.createSequentialGroup()
                                    .addGap(50)
                                    .addComponent(l2,
GroupLayout.PREFERRED_SIZE, 250, GroupLayout.PREFERRED_SIZE)
```

```
.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED))

.addGroup(contentPaneLayout.createSequentialGroup()
                                .addGap(34)
                                .addComponent(sp,
GroupLayout.PREFERRED_SIZE, 170, GroupLayout.PREFERRED_SIZE)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)
                                .addGap(34)
                                .addComponent(ttt,
GroupLayout.PREFERRED_SIZE, 170, GroupLayout.PREFERRED_SIZE)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED))
                              .addGap(14)

.addGroup(contentPaneLayout.createSequentialGroup()
                                .addGap(34)
                                .addComponent(ta,
GroupLayout.PREFERRED_SIZE, 170, GroupLayout.PREFERRED_SIZE)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED))
                              .addGap(90)

.addGroup(contentPaneLayout.createSequentialGroup()
                                .addGap(99)
                                .addComponent(l3,
GroupLayout.PREFERRED_SIZE, 150, GroupLayout.PREFERRED_SIZE)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)))
```

```
                        .addContainerGap(11, Short.MAX_VALUE))
        );
        contentPaneLayout.setVerticalGroup(
            contentPaneLayout.createParallelGroup()
                .addGroup(contentPaneLayout.createSequentialGroup()
                        .addContainerGap()

.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)
                                .addGap(31)
                                .addComponent(l1,
GroupLayout.PREFERRED_SIZE, 60, GroupLayout.PREFERRED_SIZE)
                                .addGap(69)
                                .addComponent(ex,
GroupLayout.PREFERRED_SIZE, 60, GroupLayout.PREFERRED_SIZE))
                        .addGap(39)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)

.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)
                                .addGap(50)
                                .addComponent(l2,
GroupLayout.PREFERRED_SIZE, 55, GroupLayout.PREFERRED_SIZE))

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)

.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)
                                .addGap(34)
```

```
                                .addComponent(sp,
GroupLayout.PREFERRED_SIZE, 55, GroupLayout.PREFERRED_SIZE)

                                .addGap(34)

                                .addComponent(ttt,
GroupLayout.PREFERRED_SIZE, 55, GroupLayout.PREFERRED_SIZE))

                            .addGap(14)


.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)


.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)

                                .addGap(34)

                                .addComponent(ta,
GroupLayout.PREFERRED_SIZE, 55, GroupLayout.PREFERRED_SIZE))

                            .addGap(90)


.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)

                                .addGap(99)

                                .addComponent(l3,
GroupLayout.PREFERRED_SIZE, 55, GroupLayout.PREFERRED_SIZE))

                            .addContainerGap(15, Short.MAX_VALUE))

        );

        pack();

        setLocationRelativeTo(getOwner());

    }


    JButton sp, ttt, ta, ex;
```

```
    JLabel l1, l2, l3;

}
```

```
import java.awt.Color;
import java.io.*;
import java.util.Random;
import java.util.Scanner;
import java.util.concurrent.TimeUnit;

public class Main_SpacePiano {

    public static boolean correct = false;

    private static SpacePiano board = new SpacePiano();

    private static Scanner sc = new Scanner(System.in);

    private static Random rndm = new Random();

    private static TimeUnit Unit = TimeUnit.MILLISECONDS;

    private static long[] highest = {0};

    public static int count = 3000;

    public static String l1 = "", l3 = "", t1 = "", t2 = "";
```

```
private static long score = -1;

public static void main(String[] args) {
    score = -1;
    board.setBackground(Color.white);
    board.a1.getParent().setBackground(Color.white);
    board.a2.getParent().setBackground(Color.white);
    board.a3.getParent().setBackground(Color.white);
    board.a4.getParent().setBackground(Color.white);
    if ( board.isVisible() == false ) {
        set_start();
        board.setVisible(true);
        set_start();
        board.l2.setText("0");
        board.t2.setText("Start");
        board.toFront();
    }

}

public static void main() {
    score = -1;
    board.setBackground(Color.white);
    board.a1.getParent().setBackground(Color.white);
    board.a2.getParent().setBackground(Color.white);
    board.a3.getParent().setBackground(Color.white);
```

```
        board.a4.getParent().setBackground(Color.white);
        if ( board.isVisible() == false ) {
            set_start();
            board.setVisible(true);
            set_start();
            board.l2.setText("0");
            count = 3000;
            board.t2.setText("Start");
            board.toFront();
        }
    }

    public static void time() {
        count -= 2;
        if ( count == 0 ) {
            board.t2.setText("Out of Time");
        }
        if ( count > 0 ) {
            int misec = count % 1000;
            int sec = count / 1000;
            int min = sec / 60;
            board.t2.setText(min + " : " + sec + " . " + misec);
        }
    }

    public static void set_start() {
```

```
int i;
i = rndm.nextInt(3);
switch ( i ) {
    case 0:
        board.d1.setBackground(Color.black);
        board.d2.setBackground(Color.white);
        board.d3.setBackground(Color.white);
        board.d4.setBackground(Color.white);
        break;
    case 2:
        board.d1.setBackground(Color.white);
        board.d2.setBackground(Color.black);
        board.d3.setBackground(Color.white);
        board.d4.setBackground(Color.white);
        break;
    case 3:
        board.d1.setBackground(Color.white);
        board.d2.setBackground(Color.white);
        board.d3.setBackground(Color.black);
        board.d4.setBackground(Color.white);
        break;
    case 1:
        board.d1.setBackground(Color.white);
        board.d2.setBackground(Color.white);
        board.d3.setBackground(Color.white);
        board.d4.setBackground(Color.black);
```

```
        break;
    }
    i = rndm.nextInt(3);
    switch ( i ) {
        case 1:
            board.c1.setBackground(Color.black);
            board.c2.setBackground(Color.white);
            board.c3.setBackground(Color.white);
            board.c4.setBackground(Color.white);
            break;
        case 2:
            board.c1.setBackground(Color.white);
            board.c2.setBackground(Color.black);
            board.c3.setBackground(Color.white);
            board.c4.setBackground(Color.white);
            break;
        case 3:
            board.c1.setBackground(Color.white);
            board.c2.setBackground(Color.white);
            board.c3.setBackground(Color.black);
            board.c4.setBackground(Color.white);
            break;
        case 0:
            board.c1.setBackground(Color.white);
            board.c2.setBackground(Color.white);
            board.c3.setBackground(Color.white);
```

```
        board.c4.setBackground(Color.black);
        break;
}
i = rndm.nextInt(3);
switch ( i ) {
    case 1:
        board.b1.setBackground(Color.black);
        board.b2.setBackground(Color.white);
        board.b3.setBackground(Color.white);
        board.b4.setBackground(Color.white);
        break;
    case 2:
        board.b1.setBackground(Color.white);
        board.b2.setBackground(Color.black);
        board.b3.setBackground(Color.white);
        board.b4.setBackground(Color.white);
        break;
    case 3:
        board.b1.setBackground(Color.white);
        board.b2.setBackground(Color.white);
        board.b3.setBackground(Color.black);
        board.b4.setBackground(Color.white);
        break;
    case 0:
        board.b1.setBackground(Color.white);
        board.b2.setBackground(Color.white);
```

```
        board.b3.setBackground(Color.white);
        board.b4.setBackground(Color.black);
        break;
}
i = rndm.nextInt(3);
switch ( i ) {
    case 1:
        board.a1.setBackground(Color.black);
        board.a2.setBackground(Color.white);
        board.a3.setBackground(Color.white);
        board.a4.setBackground(Color.white);
        break;
    case 2:
        board.a1.setBackground(Color.white);
        board.a2.setBackground(Color.black);
        board.a3.setBackground(Color.white);
        board.a4.setBackground(Color.white);
        break;
    case 3:
        board.a1.setBackground(Color.white);
        board.a2.setBackground(Color.white);
        board.a3.setBackground(Color.black);
        board.a4.setBackground(Color.white);
        break;
    case 0:
        board.a1.setBackground(Color.white);
```

```
            board.a2.setBackground(Color.white);
            board.a3.setBackground(Color.white);
            board.a4.setBackground(Color.black);
            break;
        }

    }

    private static void set_after() {
        int i;
        board.a1.setBackground(board.b1.getBackground());
        board.a2.setBackground(board.b2.getBackground());
        board.a3.setBackground(board.b3.getBackground());
        board.a4.setBackground(board.b4.getBackground());

        board.b1.setBackground(board.c1.getBackground());
        board.b2.setBackground(board.c2.getBackground());
        board.b3.setBackground(board.c3.getBackground());
        board.b4.setBackground(board.c4.getBackground());

        board.c1.setBackground(board.d1.getBackground());
        board.c2.setBackground(board.d2.getBackground());
        board.c3.setBackground(board.d3.getBackground());
        board.c4.setBackground(board.d4.getBackground());

        i = rndm.nextInt(3);
```

```
switch ( i ) {
    case 1:
        board.d1.setBackground(Color.black);
        board.d2.setBackground(Color.white);
        board.d3.setBackground(Color.white);
        board.d4.setBackground(Color.white);
        break;
    case 2:
        board.d1.setBackground(Color.white);
        board.d2.setBackground(Color.black);
        board.d3.setBackground(Color.white);
        board.d4.setBackground(Color.white);
        break;
    case 3:
        board.d1.setBackground(Color.white);
        board.d2.setBackground(Color.white);
        board.d3.setBackground(Color.black);
        board.d4.setBackground(Color.white);
        break;
    case 0:
        board.d1.setBackground(Color.white);
        board.d2.setBackground(Color.white);
        board.d3.setBackground(Color.white);
        board.d4.setBackground(Color.black);
        break;
}
```

```
}

public static void check() throws InterruptedException {
    if ( correct = true ) {
        set_after();
        correct = false;
        score++;
        board.l2.setText(Long.toString(score + 1));
        if ( score <= 80 && score > 0 ) {
            count = 3000;
        }
        if ( score > 80 && score <= 150 ) {
            count = 2500;
        }
        if ( score > 150 && score <= 235 ) {
            count = 2100;
        }
        if ( score > 235 && score <= 300 ) {
            count = 2000;
        }
        if ( score > 300 && score <= 399 ) {
            count = 1800;
        }
        if ( score > 399 && score <= 480 ) {
            count = 1500;
        }
```

```
            if ( score > 480 ) {
                count = 1200;
            }
            board.timer.restart();
        }
        if ( board.a1.getParent().getBackground().equals(Color.red) ||
board.a2.getParent().getBackground().equals(Color.red) ||
board.a3.getParent().getBackground().equals(Color.red) ||
board.a4.getParent().getBackground().equals(Color.red) ||
board.t2.getText().equals("Out of Time") ) {
            Unit.sleep(2337);
            board.timer.stop();
            board.dispose();
            SP_Scores b = new SP_Scores();
            if ( board.t2.getText().equals("Out of Time") ) {
                l1 = ("Time Out");
            } else {
                l1 = ("You have clicked wrong tile");
            }
            l3 = (Long.toString(score));
            try {
                FileInputStream fis = new FileInputStream("ab.txt");
                ObjectInputStream ois = new ObjectInputStream(fis);
                highest = (long[]) ois.readObject();
                ois.close();
            } catch ( Exception e ) {
            }
```

```
            if ( highest[0] < score ) {
                t1 = ("New highest scores.");
                highest[0] = score;
                try {
                    FileOutputStream fos = new FileOutputStream("ab.txt");
                    ObjectOutputStream oos = new ObjectOutputStream(fos);
                    oos.writeObject(highest);
                    oos.close();
                } catch ( Exception e ) {
                }
            } else if ( highest[0] > score ) {
                t1 = ("Highest Scores are :- ");
                t2 = (Long.toString(highest[0]));
            } else if ( highest[0] == score ) {
                t1 = ("Scores similar to Highest Scores");
            }
            Main.exit_a = true;
        }
    }

}
```

```
import java.awt.Color;
import java.awt.Container;
import java.awt.Cursor;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.*;

public class SpacePiano extends JFrame {

    private static Main_SpacePiano main = new Main_SpacePiano();

    private static SpacePiano sp = new SpacePiano();

    SpacePiano() {
        initComponents();
    }

    public static Timer timer = new Timer(2, new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            Main_SpacePiano.time();
        }
    });
```

```
void d1ActionPerformed(ActionEvent e) {
}

void d2ActionPerformed(ActionEvent e) {
}

void d3ActionPerformed(ActionEvent e) {
}

void d4ActionPerformed(ActionEvent e) {
}

void c1ActionPerformed(ActionEvent e) {
}

void c2ActionPerformed(ActionEvent e) {
}

void c3ActionPerformed(ActionEvent e) {
}

void c4ActionPerformed(ActionEvent e) {
}

void b1ActionPerformed(ActionEvent e) {
```

```
    }

    void b2ActionPerformed(ActionEvent e) {
    }

    void b3ActionPerformed(ActionEvent e) {
    }

    void b4ActionPerformed(ActionEvent e) {
    }

    void a1ActionPerformed(ActionEvent e) throws InterruptedException {
        if ( !(a1.getParent().getBackground().equals(Color.red) &&
sp.t2.getText().equals("Out of Time")) ) {
            if ( a1.getBackground().equals(Color.black) ) {
                Main_SpacePiano.correct = true;
            } else {
                Main_SpacePiano.correct = false;
                a1.getParent().setBackground(Color.red);
                a1.setBackground(Color.red);
            }
            Main_SpacePiano.check();
        }
    }

    void a2ActionPerformed(ActionEvent e) throws InterruptedException {
```

```
        if ( !(a2.getParent().getBackground().equals(Color.red) &&
sp.t2.getText().equals("Out of Time")) ) {
            if ( a2.getBackground().equals(Color.black) ) {
                Main_SpacePiano.correct = true;
            } else {
                Main_SpacePiano.correct = false;
                a2.getParent().setBackground(Color.red);
                a2.setBackground(Color.red);
            }
            Main_SpacePiano.check();
        }
    }

    void a3ActionPerformed(ActionEvent e) throws InterruptedException {
        if ( !(a3.getParent().getBackground().equals(Color.red) &&
sp.t2.getText().equals("Out of Time")) ) {
            if ( a3.getBackground().equals(Color.black) ) {
                Main_SpacePiano.correct = true;
            } else {
                Main_SpacePiano.correct = false;
                a3.getParent().setBackground(Color.red);
                a3.setBackground(Color.red);
            }
            Main_SpacePiano.check();
        }
    }
```

```
    void a4ActionPerformed(ActionEvent e) throws InterruptedException {
        if ( !(a4.getParent().getBackground().equals(Color.red) &&
sp.t2.getText().equals("Out of Time")) ) {
            if ( a4.getBackground().equals(Color.black) ) {
                Main_SpacePiano.correct = true;
            } else {
                Main_SpacePiano.correct = false;
                a4.getParent().setBackground(Color.red);
                a4.setBackground(Color.red);
            }
            Main_SpacePiano.check();
        }
    }

    void initComponents() {
        d1 = new JButton();
        d2 = new JButton();
        d3 = new JButton();
        d4 = new JButton();
        c1 = new JButton();
        c2 = new JButton();
        c3 = new JButton();
        c4 = new JButton();
        b1 = new JButton();
        b2 = new JButton();
```

```
        b3 = new JButton();
        b4 = new JButton();
        a1 = new JButton();
        a2 = new JButton();
        a3 = new JButton();
        a4 = new JButton();
        l1 = new JLabel();
        l2 = new JLabel();
        t1 = new JLabel();
        t2 = new JLabel();


setDefaultCloseOperation(javax.swing.WindowConstants.DO_NOTHING_
ON_CLOSE);
        setCursor(Cursor.HAND_CURSOR);
        setResizable(false);
        setAlwaysOnTop(true);

        setTitle("Space Piano");
        Container contentPane = getContentPane();

        l1.setText("Your Scores are : ");
        t1.setText("Time left is : ");
        t2.setText("Start");

        d1.setBackground(Color.white);
```

```
d1.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        d1ActionPerformed(e);
    }
});
d2.setBackground(Color.white);
d2.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        d2ActionPerformed(e);
    }
});
d3.setBackground(Color.white);
d3.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        d3ActionPerformed(e);
    }
});
d4.setBackground(Color.white);
d4.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        d4ActionPerformed(e);
    }
```

```
});
c1.setBackground(Color.white);
c1.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        c1ActionPerformed(e);
    }
});
c2.setBackground(Color.white);
c2.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        c2ActionPerformed(e);
    }
});
c3.setBackground(Color.white);
c3.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        c3ActionPerformed(e);
    }
});
c4.setBackground(Color.white);
c4.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
```

```
        c4ActionPerformed(e);
    }
});
b1.setBackground(Color.white);
b1.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        b1ActionPerformed(e);
    }
});
b2.setBackground(Color.white);
b2.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        b2ActionPerformed(e);
    }
});
b3.setBackground(Color.white);
b3.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        b3ActionPerformed(e);
    }
});
b4.setBackground(Color.white);
b4.addActionListener(new ActionListener() {
```

```
            @Override
            public void actionPerformed(ActionEvent e) {
                b4ActionPerformed(e);
            }
        });
        a1.setBackground(Color.white);
        a1.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                try {
                    a1ActionPerformed(e);
                } catch ( InterruptedException ex ) {

Logger.getLogger(SpacePiano.class.getName()).log(Level.SEVERE, null,
ex);
                }
            }
        });
        a2.setBackground(Color.white);
        a2.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                try {
                    a2ActionPerformed(e);
                } catch ( InterruptedException ex ) {
```

```
Logger.getLogger(SpacePiano.class.getName()).log(Level.SEVERE, null,
ex);
            }
        }
    });
    a3.setBackground(Color.white);
    a3.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            try {
                a3ActionPerformed(e);
            } catch ( InterruptedException ex ) {

Logger.getLogger(SpacePiano.class.getName()).log(Level.SEVERE, null,
ex);
            }
        }
    });
    a4.setBackground(Color.white);
    a4.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            try {
                a4ActionPerformed(e);
            } catch ( InterruptedException ex ) {
```

```
Logger.getLogger(SpacePiano.class.getName()).log(Level.SEVERE, null,
ex);
            }
        }
    });

    GroupLayout contentPaneLayout = new GroupLayout(contentPane);
    contentPane.setLayout(contentPaneLayout);
    contentPaneLayout.setHorizontalGroup(
        contentPaneLayout.createParallelGroup()
            .addGroup(contentPaneLayout.createSequentialGroup()
                .addContainerGap()
                .addGroup(contentPaneLayout.createParallelGroup()

.addGroup(contentPaneLayout.createSequentialGroup()
                        .addComponent(d1,
GroupLayout.PREFERRED_SIZE, 95, GroupLayout.PREFERRED_SIZE)
                        .addComponent(d2,
GroupLayout.PREFERRED_SIZE, 95, GroupLayout.PREFERRED_SIZE)
                        .addComponent(d3,
GroupLayout.PREFERRED_SIZE, 95, GroupLayout.PREFERRED_SIZE)
                        .addComponent(d4,
GroupLayout.PREFERRED_SIZE, 95, GroupLayout.PREFERRED_SIZE))

.addGroup(contentPaneLayout.createSequentialGroup()
                        .addComponent(c1,
GroupLayout.PREFERRED_SIZE, 95, GroupLayout.PREFERRED_SIZE)
```

```
                                .addComponent(c2,
GroupLayout.PREFERRED_SIZE, 95, GroupLayout.PREFERRED_SIZE)

                                .addComponent(c3,
GroupLayout.PREFERRED_SIZE, 95, GroupLayout.PREFERRED_SIZE)

                                .addComponent(c4,
GroupLayout.PREFERRED_SIZE, 95, GroupLayout.PREFERRED_SIZE))

.addGroup(contentPaneLayout.createSequentialGroup()

                                .addComponent(b1,
GroupLayout.PREFERRED_SIZE, 95, GroupLayout.PREFERRED_SIZE)

                                .addComponent(b2,
GroupLayout.PREFERRED_SIZE, 95, GroupLayout.PREFERRED_SIZE)

                                .addComponent(b3,
GroupLayout.PREFERRED_SIZE, 95, GroupLayout.PREFERRED_SIZE)

                                .addComponent(b4,
GroupLayout.PREFERRED_SIZE, 95, GroupLayout.PREFERRED_SIZE))

.addGroup(contentPaneLayout.createSequentialGroup()

                                .addComponent(a1,
GroupLayout.PREFERRED_SIZE, 95, GroupLayout.PREFERRED_SIZE)

                                .addComponent(a2,
GroupLayout.PREFERRED_SIZE, 95, GroupLayout.PREFERRED_SIZE)

                                .addComponent(a3,
GroupLayout.PREFERRED_SIZE, 95, GroupLayout.PREFERRED_SIZE)

                                .addComponent(a4,
GroupLayout.PREFERRED_SIZE, 95, GroupLayout.PREFERRED_SIZE))

.addGroup(contentPaneLayout.createSequentialGroup()

                                .addComponent(l1,
GroupLayout.PREFERRED_SIZE, 150, GroupLayout.PREFERRED_SIZE)

                                .addComponent(l2,
GroupLayout.PREFERRED_SIZE, 30, GroupLayout.PREFERRED_SIZE))
```

```
.addGroup(contentPaneLayout.createSequentialGroup()

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)
                                    .addComponent(t1,
GroupLayout.PREFERRED_SIZE, 150, GroupLayout.PREFERRED_SIZE)
                                    .addComponent(t2,
GroupLayout.PREFERRED_SIZE, 100,
GroupLayout.PREFERRED_SIZE)))
                        .addContainerGap(11, Short.MAX_VALUE))
        );
        contentPaneLayout.setVerticalGroup(
            contentPaneLayout.createParallelGroup()
                .addGroup(contentPaneLayout.createSequentialGroup()
                    .addContainerGap()

.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)
                                .addComponent(d1,
GroupLayout.PREFERRED_SIZE, 138, GroupLayout.PREFERRED_SIZE)
                                .addComponent(d2,
GroupLayout.PREFERRED_SIZE, 138, GroupLayout.PREFERRED_SIZE)
                                .addComponent(d3,
GroupLayout.PREFERRED_SIZE, 138, GroupLayout.PREFERRED_SIZE)
                                .addComponent(d4,
GroupLayout.PREFERRED_SIZE, 138, GroupLayout.PREFERRED_SIZE))

.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)
```

```
                    .addComponent(c1,
GroupLayout.PREFERRED_SIZE, 138, GroupLayout.PREFERRED_SIZE)

                    .addComponent(c2,
GroupLayout.PREFERRED_SIZE, 138, GroupLayout.PREFERRED_SIZE)

                    .addComponent(c3,
GroupLayout.PREFERRED_SIZE, 138, GroupLayout.PREFERRED_SIZE)

                    .addComponent(c4,
GroupLayout.PREFERRED_SIZE, 138, GroupLayout.PREFERRED_SIZE))

.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)

                    .addComponent(b1,
GroupLayout.PREFERRED_SIZE, 138, GroupLayout.PREFERRED_SIZE)

                    .addComponent(b2,
GroupLayout.PREFERRED_SIZE, 138, GroupLayout.PREFERRED_SIZE)

                    .addComponent(b3,
GroupLayout.PREFERRED_SIZE, 138, GroupLayout.PREFERRED_SIZE)

                    .addComponent(b4,
GroupLayout.PREFERRED_SIZE, 138, GroupLayout.PREFERRED_SIZE))

.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)

                    .addComponent(a1,
GroupLayout.PREFERRED_SIZE, 138, GroupLayout.PREFERRED_SIZE)

                    .addComponent(a2,
GroupLayout.PREFERRED_SIZE, 138, GroupLayout.PREFERRED_SIZE)

                    .addComponent(a3,
GroupLayout.PREFERRED_SIZE, 138, GroupLayout.PREFERRED_SIZE)

                    .addComponent(a4,
GroupLayout.PREFERRED_SIZE, 138, GroupLayout.PREFERRED_SIZE))
```

```
.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)
                                .addComponent(l1,
GroupLayout.PREFERRED_SIZE, 40, GroupLayout.PREFERRED_SIZE)
                                .addComponent(l2,
GroupLayout.PREFERRED_SIZE, 40, GroupLayout.PREFERRED_SIZE))

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)

.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)
                                .addComponent(t1,
GroupLayout.PREFERRED_SIZE, 40, GroupLayout.PREFERRED_SIZE)
                                .addComponent(t2,
GroupLayout.PREFERRED_SIZE, 40, GroupLayout.PREFERRED_SIZE))
                            .addContainerGap(15, Short.MAX_VALUE))
        );
        pack();
        setLocationRelativeTo(getOwner());
    }



    JButton d1, d2, d3, d4;


    JButton c1, c2, c3, c4;
```

```
    JButton b1, b2, b3, b4;

    JButton a1, a2, a3, a4;

    JLabel l1, l2, t1, t2;

}
```

```
import java.awt.Container;
import java.awt.Cursor;
import javax.swing.*;

public class SP_Scores extends JFrame {

    SP_Scores() {
        initComponents();
    }

    void initComponents() {
        l1 = new JLabel();
        l2 = new JLabel();
        l3 = new JLabel();
        t1 = new JLabel();
        t2 = new JLabel();


setDefaultCloseOperation(javax.swing.WindowConstants.DO_NOTHING_
ON_CLOSE);
        setCursor(Cursor.HAND_CURSOR);
        setResizable(false);

        setTitle("Space Piano Scores");
```

```
        Container contentPane = getContentPane();

        l2.setText("Your Scores are : ");
        l1.setText(Main_SpacePiano.l1);
        l3.setText(Main_SpacePiano.l3);
        t1.setText(Main_SpacePiano.t1);
        t2.setText(Main_SpacePiano.t2);

        GroupLayout contentPaneLayout = new GroupLayout(contentPane);
        contentPane.setLayout(contentPaneLayout);
        contentPaneLayout.setHorizontalGroup(
                contentPaneLayout.createParallelGroup()
                        .addGroup(contentPaneLayout.createSequentialGroup()
                                .addContainerGap()
                                .addGroup(contentPaneLayout.createParallelGroup()

.addGroup(contentPaneLayout.createSequentialGroup()
                                        .addGap(21)
                                        .addComponent(l1,
GroupLayout.PREFERRED_SIZE, 230, GroupLayout.PREFERRED_SIZE)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED))
                                .addGap(7)

.addGroup(contentPaneLayout.createSequentialGroup()
                                        .addComponent(l2,
GroupLayout.PREFERRED_SIZE, 105, GroupLayout.PREFERRED_SIZE)
```

```
.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)
                                        .addGap(10)
                                        .addComponent(l3,
GroupLayout.PREFERRED_SIZE, 55, GroupLayout.PREFERRED_SIZE)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED))

.addGroup(contentPaneLayout.createSequentialGroup()
                                        .addComponent(t1,
GroupLayout.PREFERRED_SIZE, 255, GroupLayout.PREFERRED_SIZE)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)
                                        .addGap(10)
                                        .addComponent(t2,
GroupLayout.PREFERRED_SIZE, 55, GroupLayout.PREFERRED_SIZE)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)))
                                .addContainerGap(11, Short.MAX_VALUE))
        );
        contentPaneLayout.setVerticalGroup(
                contentPaneLayout.createParallelGroup()
                        .addGroup(contentPaneLayout.createSequentialGroup()
                                .addContainerGap()

.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)
                                        .addGap(21)
                                        .addComponent(l1,
GroupLayout.PREFERRED_SIZE, 138, GroupLayout.PREFERRED_SIZE))
```

```
                        .addGap(7)

.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)

                                .addComponent(l2,
GroupLayout.PREFERRED_SIZE, 138, GroupLayout.PREFERRED_SIZE)

                                .addGap(10)

                                .addComponent(l3,
GroupLayout.PREFERRED_SIZE, 138, GroupLayout.PREFERRED_SIZE))


.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)

                                .addComponent(t1,
GroupLayout.PREFERRED_SIZE, 138, GroupLayout.PREFERRED_SIZE)

                                .addGap(10)

                                .addComponent(t2,
GroupLayout.PREFERRED_SIZE, 138, GroupLayout.PREFERRED_SIZE))

                        .addContainerGap(15, Short.MAX_VALUE))

        );

        pack();

        setLocationRelativeTo(getOwner());

    }


    JLabel l1, l2, l3, t1, t2;


}
```

```
import java.util.Scanner;

public class Main_tictactoe {

    public static boolean player_a_turn = true, player_b_won = false,
player_a_won = false;

    public static boolean draw = false, play_again = false;

    private static byte firstTurn = 0;

    public static String won = "";

    private static Scanner sc = new Scanner(System.in);

    public static Tictactoe board = new Tictactoe();

    private static Main main = new Main();

    public static void main(String[] args) {
        if ( board.isVisible() == false ) {
            board.setVisible(true);
            board.toFront();
        }
```

```
}

public static void main() {
    board.button1.setText("");
    board.button2.setText("");
    board.button3.setText("");
    board.button4.setText("");
    board.button5.setText("");
    board.button6.setText("");
    board.button7.setText("");
    board.button8.setText("");
    board.button9.setText("");
    player_a_turn = true;
    player_b_won = false;
    player_a_won = false;
    draw = false;
    play_again = false;
    won = "";
    firstTurn = 0;
    if ( board.isVisible() == false ) {
        board.setVisible(true);
        board.toFront();
    }
}

public static void check() {
```

```
if ( board.button1.getText().equals("X") ) {
    if ( board.button4.getText().equals("X") ) {
        if ( board.button7.getText().equals("X") ) {
            player_a_won = true;
            player_b_won = false;
            won = "Player X has won";
        }
    }
}
if ( board.button1.getText().equals("X") ) {
    if ( board.button5.getText().equals("X") ) {
        if ( board.button9.getText().equals("X") ) {
            player_a_won = true;
            player_b_won = false;
            won = "Player X has won";
        }
    }
}
if ( board.button1.getText().equals("X") ) {
    if ( board.button2.getText().equals("X") ) {
        if ( board.button3.getText().equals("X") ) {
            player_a_won = true;
            player_b_won = false;
            won = "Player X has won";
        }
    }
```

```
}
if ( board.button3.getText().equals("X") ) {
    if ( board.button5.getText().equals("X") ) {
        if ( board.button7.getText().equals("X") ) {
            player_a_won = true;
            player_b_won = false;
            won = "Player X has won";
        }
    }
}
if ( board.button3.getText().equals("X") ) {
    if ( board.button6.getText().equals("X") ) {
        if ( board.button9.getText().equals("X") ) {
            player_a_won = true;
            player_b_won = false;
            won = "Player X has won";
        }
    }
}
if ( board.button7.getText().equals("X") ) {
    if ( board.button8.getText().equals("X") ) {
        if ( board.button9.getText().equals("X") ) {
            player_a_won = true;
            player_b_won = false;
            won = "Player X has won";
        }
```

```
    }
}
if ( board.button4.getText().equals("X") ) {
    if ( board.button5.getText().equals("X") ) {
        if ( board.button6.getText().equals("X") ) {
            player_a_won = true;
            player_b_won = false;
            won = "Player X has won";
        }
    }
}
if ( board.button2.getText().equals("X") ) {
    if ( board.button5.getText().equals("X") ) {
        if ( board.button8.getText().equals("X") ) {
            player_a_won = true;
            player_b_won = false;
            won = "Player X has won";
        }
    }
}

if ( board.button1.getText().equals("O") ) {
    if ( board.button4.getText().equals("O") ) {
        if ( board.button7.getText().equals("O") ) {
            player_a_won = false;
            player_b_won = true;
```

```
            won = "Player O has won";
        }
    }
}
if ( board.button1.getText().equals("O") ) {
    if ( board.button5.getText().equals("O") ) {
        if ( board.button9.getText().equals("O") ) {
            player_a_won = false;
            player_b_won = true;
            won = "Player O has won";
        }
    }
}
if ( board.button1.getText().equals("O") ) {
    if ( board.button2.getText().equals("O") ) {
        if ( board.button3.getText().equals("O") ) {
            player_a_won = false;
            player_b_won = true;
            won = "Player O has won";
        }
    }
}
if ( board.button3.getText().equals("O") ) {
    if ( board.button5.getText().equals("O") ) {
        if ( board.button7.getText().equals("O") ) {
            player_a_won = false;
```

```
            player_b_won = true;
            won = "Player O has won";
        }
    }
}
if ( board.button3.getText().equals("O") ) {
    if ( board.button6.getText().equals("O") ) {
        if ( board.button9.getText().equals("O") ) {
            player_a_won = false;
            player_b_won = true;
            won = "Player O has won";
        }
    }
}
if ( board.button7.getText().equals("O") ) {
    if ( board.button8.getText().equals("O") ) {
        if ( board.button9.getText().equals("O") ) {
            player_a_won = false;
            player_b_won = true;
            won = "Player O has won";
        }
    }
}
if ( board.button4.getText().equals("O") ) {
    if ( board.button5.getText().equals("O") ) {
        if ( board.button6.getText().equals("O") ) {
```

```
                player_a_won = false;
                player_b_won = true;
                won = "Player O has won";
            }
        }
    }
    if ( board.button2.getText().equals("O") ) {
        if ( board.button5.getText().equals("O") ) {
            if ( board.button8.getText().equals("O") ) {
                player_a_won = false;
                player_b_won = true;
                won = "Player O has won";
            }
        }
    }
    if ( (board.button1.getText().equals("X")) || 
(board.button1.getText().equals("O")) ) {
        if ( (board.button2.getText().equals("X")) || 
(board.button2.getText().equals("O")) ) {
            if ( (board.button3.getText().equals("X")) || 
(board.button3.getText().equals("O")) ) {
                if ( (board.button4.getText().equals("X")) || 
(board.button4.getText().equals("O")) ) {
                    if ( (board.button5.getText().equals("X")) || 
(board.button5.getText().equals("O")) ) {
                        if ( (board.button6.getText().equals("X")) || 
(board.button6.getText().equals("O")) ) {
```

```
                    if ( (board.button7.getText().equals("X")) || (board.button7.getText().equals("O")) ) {
                        if ( (board.button8.getText().equals("X")) || (board.button8.getText().equals("O")) ) {
                            if ( (board.button9.getText().equals("X")) || (board.button9.getText().equals("O")) ) {
                                draw = true;
                                won = "The Match has drawn";
                            }
                        }
                    }
                }
            }
        }
    }
    if ( player_b_won == true || player_a_won == true || draw == true ) {
        board.setVisible(false);
        Main.exit_a = true;
    }
  }
}
```

```
import java.awt.Container;
import java.awt.Cursor;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;

public class Tictactoe extends JFrame {

    public static Main_tictactoe main = new Main_tictactoe();

    Tictactoe() {
        initComponents();
    }

    void button1ActionPerformed(ActionEvent e) {
        if ( button1.getText().equals("") ) {
            if ( Main_tictactoe.player_a_turn == true ) {
                button1.setText("X");
                Main_tictactoe.check();
                Main_tictactoe.player_a_turn = false;
            } else {
                button1.setText("O");
                Main_tictactoe.check();
                Main_tictactoe.player_a_turn = true;
```

```
        }
    }
}

void button2ActionPerformed(ActionEvent e) {
    if ( button2.getText().equals("") ) {
        if ( Main_tictactoe.player_a_turn == true ) {
            button2.setText("X");
            Main_tictactoe.check();
            Main_tictactoe.player_a_turn = false;
        } else {
            button2.setText("O");
            Main_tictactoe.check();
            Main_tictactoe.player_a_turn = true;
        }
    }
}

void button3ActionPerformed(ActionEvent e) {
    if ( button3.getText().equals("") ) {
        if ( Main_tictactoe.player_a_turn == true ) {
            button3.setText("X");
            Main_tictactoe.check();
            Main_tictactoe.player_a_turn = false;
        } else {
            button3.setText("O");
```

```
            Main_tictactoe.check();
            Main_tictactoe.player_a_turn = true;
        }
    }
}

void button4ActionPerformed(ActionEvent e) {
    if ( button4.getText().equals("") ) {
        if ( Main_tictactoe.player_a_turn == true ) {
            button4.setText("X");
            Main_tictactoe.check();
            Main_tictactoe.player_a_turn = false;
        } else {
            button4.setText("O");
            Main_tictactoe.check();
            Main_tictactoe.player_a_turn = true;
        }
    }
}

void button5ActionPerformed(ActionEvent e) {
    if ( button5.getText().equals("") ) {
        if ( Main_tictactoe.player_a_turn == true ) {
            button5.setText("X");
            Main_tictactoe.check();
            Main_tictactoe.player_a_turn = false;
```

```
        } else {
            button5.setText("O");
            Main_tictactoe.check();
            Main_tictactoe.player_a_turn = true;
        }
    }
}

void button6ActionPerformed(ActionEvent e) {
    if ( button6.getText().equals("") ) {
        if ( Main_tictactoe.player_a_turn == true ) {
            button6.setText("X");
            Main_tictactoe.check();
            Main_tictactoe.player_a_turn = false;
        } else {
            button6.setText("O");
            Main_tictactoe.check();
            Main_tictactoe.player_a_turn = true;
        }
    }
}

void button7ActionPerformed(ActionEvent e) {
    if ( button7.getText().equals("") ) {
        if ( Main_tictactoe.player_a_turn == true ) {
            button7.setText("X");
```

```
            Main_tictactoe.check();
            Main_tictactoe.player_a_turn = false;
        } else {
            button7.setText("O");
            Main_tictactoe.check();
            Main_tictactoe.player_a_turn = true;
        }
    }
}

void button8ActionPerformed(ActionEvent e) {
    if ( button8.getText().equals("") ) {
        if ( Main_tictactoe.player_a_turn == true ) {
            button8.setText("X");
            Main_tictactoe.check();
            Main_tictactoe.player_a_turn = false;
        } else {
            button8.setText("O");
            Main_tictactoe.check();
            Main_tictactoe.player_a_turn = true;
        }
    }
}

void button9ActionPerformed(ActionEvent e) {
    if ( button9.getText().equals("") ) {
```

```
            if ( Main_tictactoe.player_a_turn == true ) {
                button9.setText("X");
                Main_tictactoe.check();
                Main_tictactoe.player_a_turn = false;
            } else {
                button9.setText("O");
                Main_tictactoe.check();
                Main_tictactoe.player_a_turn = true;
            }
        }
    }

    void initComponents() {
        button1 = new JButton();
        button2 = new JButton();
        button3 = new JButton();
        button4 = new JButton();
        button5 = new JButton();
        button6 = new JButton();
        button7 = new JButton();
        button8 = new JButton();
        button9 = new JButton();


setDefaultCloseOperation(javax.swing.WindowConstants.DO_NOTHING_
ON_CLOSE);
```

```
setCursor(Cursor.HAND_CURSOR);
setResizable(false);
setAlwaysOnTop(true);

setTitle("Tic Tac Toe");
Container contentPane = getContentPane();

button1.setText("");
button1.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        button1ActionPerformed(e);
    }
});
button2.setText("");
button2.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        button2ActionPerformed(e);
    }
});
button3.setText("");
button3.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        button3ActionPerformed(e);
```

```
    }
});
button4.setText("");
button4.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        button4ActionPerformed(e);
    }
});
button5.setText("");
button5.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        button5ActionPerformed(e);
    }
});
button6.setText("");
button6.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        button6ActionPerformed(e);
    }
});
button7.setText("");
button7.addActionListener(new ActionListener() {
    @Override
```

```
    public void actionPerformed(ActionEvent e) {
        button7ActionPerformed(e);
    }
});
button8.setText("");
button8.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        button8ActionPerformed(e);
    }
});
button9.setText("");
button9.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        button9ActionPerformed(e);
    }
});

GroupLayout contentPaneLayout = new GroupLayout(contentPane);
contentPane.setLayout(contentPaneLayout);
contentPaneLayout.setHorizontalGroup(
        contentPaneLayout.createParallelGroup()
                .addGroup(contentPaneLayout.createSequentialGroup()
                        .addContainerGap()
                        .addGroup(contentPaneLayout.createParallelGroup()
```

```
.addGroup(contentPaneLayout.createSequentialGroup()
                                    .addComponent(button1,
GroupLayout.PREFERRED_SIZE, 68, GroupLayout.PREFERRED_SIZE)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)
                                    .addComponent(button2,
GroupLayout.PREFERRED_SIZE, 68, GroupLayout.PREFERRED_SIZE)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)
                                    .addComponent(button3,
GroupLayout.PREFERRED_SIZE, 68, GroupLayout.PREFERRED_SIZE))

.addGroup(contentPaneLayout.createSequentialGroup()
                                    .addComponent(button4,
GroupLayout.PREFERRED_SIZE, 68, GroupLayout.PREFERRED_SIZE)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)
                                    .addComponent(button5,
GroupLayout.PREFERRED_SIZE, 68, GroupLayout.PREFERRED_SIZE)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)
                                    .addComponent(button6,
GroupLayout.PREFERRED_SIZE, 68, GroupLayout.PREFERRED_SIZE))

.addGroup(contentPaneLayout.createSequentialGroup()
                                    .addComponent(button7,
GroupLayout.PREFERRED_SIZE, 68, GroupLayout.PREFERRED_SIZE)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)
                                    .addComponent(button8,
GroupLayout.PREFERRED_SIZE, 68, GroupLayout.PREFERRED_SIZE)
```

```
.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)
                                    .addComponent(button9,
GroupLayout.PREFERRED_SIZE, 68, GroupLayout.PREFERRED_SIZE)))
                        .addContainerGap(11, Short.MAX_VALUE))
        );
        contentPaneLayout.setVerticalGroup(
            contentPaneLayout.createParallelGroup()
                .addGroup(contentPaneLayout.createSequentialGroup()
                        .addContainerGap()

.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)
                                .addComponent(button1,
GroupLayout.PREFERRED_SIZE, 55, GroupLayout.PREFERRED_SIZE)
                                .addComponent(button2,
GroupLayout.PREFERRED_SIZE, 55, GroupLayout.PREFERRED_SIZE)
                                .addComponent(button3,
GroupLayout.PREFERRED_SIZE, 55, GroupLayout.PREFERRED_SIZE))

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)

.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)
                                .addComponent(button4,
GroupLayout.PREFERRED_SIZE, 55, GroupLayout.PREFERRED_SIZE)
                                .addComponent(button5,
GroupLayout.PREFERRED_SIZE, 55, GroupLayout.PREFERRED_SIZE)
                                .addComponent(button6,
GroupLayout.PREFERRED_SIZE, 55, GroupLayout.PREFERRED_SIZE))
```

```
.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)

.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)
                                .addComponent(button7,
GroupLayout.PREFERRED_SIZE, 55, GroupLayout.PREFERRED_SIZE)
                                .addComponent(button8,
GroupLayout.PREFERRED_SIZE, 55, GroupLayout.PREFERRED_SIZE)
                                .addComponent(button9,
GroupLayout.PREFERRED_SIZE, 55, GroupLayout.PREFERRED_SIZE))
                            .addContainerGap(15, Short.MAX_VALUE))
        );
        pack();
        setLocationRelativeTo(getOwner());
    }

    JButton button1;

    JButton button2;

    JButton button3;

    JButton button4;

    JButton button5;

    JButton button6;
```

```
    JButton button7;

    JButton button8;

    JButton button9;
}
```

```
import java.awt.Container;
import java.awt.Cursor;
import javax.swing.*;

public class TTT_Scores extends JFrame {

    TTT_Scores() {
        initComponents();
    }

    void initComponents() {
        won = new JLabel();


setDefaultCloseOperation(javax.swing.WindowConstants.DO_NOTHING_
ON_CLOSE);
        setCursor(Cursor.HAND_CURSOR);
        setResizable(false);

        setTitle("Tic Tac Toe Scores");
        Container contentPane = getContentPane();

        won.setText(Main_tictactoe.won);
```

```
GroupLayout contentPaneLayout = new GroupLayout(contentPane);
contentPane.setLayout(contentPaneLayout);
contentPaneLayout.setHorizontalGroup(
        contentPaneLayout.createParallelGroup()
                .addGroup(contentPaneLayout.createSequentialGroup()
                        .addContainerGap()
                        .addGroup(contentPaneLayout.createParallelGroup()

.addGroup(contentPaneLayout.createSequentialGroup()
                                .addComponent(won,
GroupLayout.PREFERRED_SIZE, 230, GroupLayout.PREFERRED_SIZE)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED))
                                .addGap(10))
                        .addContainerGap(11, Short.MAX_VALUE))
);
contentPaneLayout.setVerticalGroup(
        contentPaneLayout.createParallelGroup()
                .addGroup(contentPaneLayout.createSequentialGroup()
                        .addContainerGap()

.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)
                                .addComponent(won,
GroupLayout.PREFERRED_SIZE, 60, GroupLayout.PREFERRED_SIZE))
                        .addGap(10)
                        .addContainerGap(15, Short.MAX_VALUE))
);
```

```
        pack();
        setLocationRelativeTo(getOwner());
    }

    JLabel won;
}
```

```
import java.awt.Color;
import java.io.*;
import java.util.Random;
import java.util.concurrent.TimeUnit;
import java.util.logging.Level;
import java.util.logging.Logger;

public class Main_TA {

    public static boolean correct = true;

    public static boolean exit = false;

    private static TA board = new TA();

    private static Random rndm = new Random();

    private static TimeUnit Unit = TimeUnit.MILLISECONDS;

    public static String l3 = "", t1 = "", t2 = "";

    private static long[] highest = {0};

    private static long score = 0;
```

```
public static void main() {
    score = 0;
    if ( board.isVisible() == false ) {
        set_start();
        board.setVisible(true);
        set_start();
        board.l2.setText("0");
        board.toFront();
        TA.timer.start();
    }
}

private static void set_start() {
    int i;
    i = rndm.nextInt(3);
    switch ( i ) {
        case 0:
            board.d1.setBackground(Color.black);
            board.d2.setBackground(Color.white);
            board.d3.setBackground(Color.white);
            board.d4.setBackground(Color.white);
            break;
        case 1:
            board.d1.setBackground(Color.white);
            board.d2.setBackground(Color.black);
```

```
            board.d3.setBackground(Color.white);
            board.d4.setBackground(Color.white);
            break;
        case 2:
            board.d1.setBackground(Color.white);
            board.d2.setBackground(Color.white);
            board.d3.setBackground(Color.black);
            board.d4.setBackground(Color.white);
            break;
        case 3:
            board.d1.setBackground(Color.white);
            board.d2.setBackground(Color.white);
            board.d3.setBackground(Color.white);
            board.d4.setBackground(Color.black);
            break;
    }
    i = rndm.nextInt(3);
    switch ( i ) {
        case 1:
            board.c1.setBackground(Color.black);
            board.c2.setBackground(Color.white);
            board.c3.setBackground(Color.white);
            board.c4.setBackground(Color.white);
            break;
        case 2:
            board.c1.setBackground(Color.white);
```

```
            board.c2.setBackground(Color.black);
            board.c3.setBackground(Color.white);
            board.c4.setBackground(Color.white);
            break;
        case 3:
            board.c1.setBackground(Color.white);
            board.c2.setBackground(Color.white);
            board.c3.setBackground(Color.black);
            board.c4.setBackground(Color.white);
            break;
        case 0:
            board.c1.setBackground(Color.white);
            board.c2.setBackground(Color.white);
            board.c3.setBackground(Color.white);
            board.c4.setBackground(Color.black);
            break;
    }
    board.b1.setBackground(Color.white);
    board.b2.setBackground(Color.white);
    board.b3.setBackground(Color.white);
    board.b4.setBackground(Color.white);
    board.a1.setBackground(Color.white);
    board.a2.setBackground(Color.white);
    board.a3.setBackground(Color.white);
    board.a4.setBackground(Color.white);
}
```

```
    public static void set_after() {
        int i;
        if ( board.a1.getBackground().equals(Color.black) ||
board.a2.getBackground().equals(Color.black) ||
board.a3.getBackground().equals(Color.black) ||
board.a4.getBackground().equals(Color.black) ) {
            correct = false;
            TA.timer.stop();
            board.dispose();
            try {
                Unit.sleep(2999);
            } catch ( InterruptedException ex ) {
                Logger.getLogger(Main_TA.class.getName()).log(Level.SEVERE,
null, ex);
            }
            l3 = (Long.toString(score));
            try {
                FileInputStream fis = new FileInputStream("TA.txt");
                ObjectInputStream ois = new ObjectInputStream(fis);
                highest = (long[]) ois.readObject();
                ois.close();
            } catch ( Exception e ) {
            }
            if ( highest[0] < score ) {
                t1 = ("New highest scores.");
                highest[0] = score;
```

```
            try {
                FileOutputStream fos = new FileOutputStream("TA.txt");
                ObjectOutputStream oos = new ObjectOutputStream(fos);
                oos.writeObject(highest);
                oos.close();
            } catch ( Exception e ) {
            }
        } else if ( highest[0] > score ) {
            t1 = ("Highest Scores are :- ");
            t2 = (Long.toString(highest[0]));
        } else if ( highest[0] == score ) {
            t1 = ("Scores similar to Highest Scores");
        }
        Main.exit_a = true;
    } else {
        afterAttack();
        board.a1.setBackground(board.b1.getBackground());
        board.a2.setBackground(board.b2.getBackground());
        board.a3.setBackground(board.b3.getBackground());
        board.a4.setBackground(board.b4.getBackground());

        board.b1.setBackground(board.c1.getBackground());
        board.b2.setBackground(board.c2.getBackground());
        board.b3.setBackground(board.c3.getBackground());
        board.b4.setBackground(board.c4.getBackground());
```

```
board.c1.setBackground(board.d1.getBackground());
board.c2.setBackground(board.d2.getBackground());
board.c3.setBackground(board.d3.getBackground());
board.c4.setBackground(board.d4.getBackground());
i = rndm.nextInt(3);
switch ( i ) {
    case 1:
        board.d1.setBackground(Color.black);
        board.d2.setBackground(Color.white);
        board.d3.setBackground(Color.white);
        board.d4.setBackground(Color.white);
        break;
    case 2:
        board.d1.setBackground(Color.white);
        board.d2.setBackground(Color.black);
        board.d3.setBackground(Color.white);
        board.d4.setBackground(Color.white);
        break;
    case 3:
        board.d1.setBackground(Color.white);
        board.d2.setBackground(Color.white);
        board.d3.setBackground(Color.black);
        board.d4.setBackground(Color.white);
        break;
    case 0:
        board.d1.setBackground(Color.white);
```

```
                board.d2.setBackground(Color.white);
                board.d3.setBackground(Color.white);
                board.d4.setBackground(Color.black);
                break;
            }
        }
    }

    private static void afterAttack() {

        if ( board.a1.getBackground().equals(Color.lightGray) ) {
            board.a1.setBackground(Color.white);
        } else if ( board.a2.getBackground().equals(Color.lightGray) ) {
            board.a2.setBackground(Color.white);
        } else if ( board.a3.getBackground().equals(Color.lightGray) ) {
            board.a3.setBackground(Color.white);
        } else if ( board.a4.getBackground().equals(Color.lightGray) ) {
            board.a4.setBackground(Color.white);
        }

        if ( board.b1.getBackground().equals(Color.lightGray) ) {
            board.b1.setBackground(Color.white);
        } else if ( board.b2.getBackground().equals(Color.lightGray) ) {
            board.b2.setBackground(Color.white);
        } else if ( board.b3.getBackground().equals(Color.lightGray) ) {
            board.b3.setBackground(Color.white);
```

```
} else if ( board.b4.getBackground().equals(Color.lightGray) ) {
    board.b4.setBackground(Color.white);
}

if ( board.c1.getBackground().equals(Color.lightGray) ) {
    board.c1.setBackground(Color.white);
} else if ( board.c2.getBackground().equals(Color.lightGray) ) {
    board.c2.setBackground(Color.white);
} else if ( board.c3.getBackground().equals(Color.lightGray) ) {
    board.c3.setBackground(Color.white);
} else if ( board.c4.getBackground().equals(Color.lightGray) ) {
    board.c4.setBackground(Color.white);
}

if ( board.d1.getBackground().equals(Color.lightGray) ) {
    board.d1.setBackground(Color.white);
} else if ( board.d2.getBackground().equals(Color.lightGray) ) {
    board.d2.setBackground(Color.white);
} else if ( board.d3.getBackground().equals(Color.lightGray) ) {
    board.d3.setBackground(Color.white);
} else if ( board.d4.getBackground().equals(Color.lightGray) ) {
    board.d4.setBackground(Color.white);
}
if ( board.a1.getBackground().equals(Color.cyan) ) {
    board.a1.setBackground(Color.lightGray);
} else if ( board.a2.getBackground().equals(Color.cyan) ) {
```

```
    board.a2.setBackground(Color.lightGray);
} else if ( board.a3.getBackground().equals(Color.cyan) ) {
    board.a3.setBackground(Color.lightGray);
} else if ( board.a4.getBackground().equals(Color.cyan) ) {
    board.a4.setBackground(Color.lightGray);
}

if ( board.b1.getBackground().equals(Color.cyan) ) {
    board.b1.setBackground(Color.lightGray);
} else if ( board.b2.getBackground().equals(Color.cyan) ) {
    board.b2.setBackground(Color.lightGray);
} else if ( board.b3.getBackground().equals(Color.cyan) ) {
    board.b3.setBackground(Color.lightGray);
} else if ( board.b4.getBackground().equals(Color.cyan) ) {
    board.b4.setBackground(Color.lightGray);
}

if ( board.c1.getBackground().equals(Color.cyan) ) {
    board.c1.setBackground(Color.lightGray);
} else if ( board.c2.getBackground().equals(Color.cyan) ) {
    board.c2.setBackground(Color.lightGray);
} else if ( board.c3.getBackground().equals(Color.cyan) ) {
    board.c3.setBackground(Color.lightGray);
} else if ( board.c4.getBackground().equals(Color.cyan) ) {
    board.c4.setBackground(Color.lightGray);
}
```

```
    if ( board.d1.getBackground().equals(Color.cyan) ) {
        board.d1.setBackground(Color.lightGray);
    } else if ( board.d2.getBackground().equals(Color.cyan) ) {
        board.d2.setBackground(Color.lightGray);
    } else if ( board.d3.getBackground().equals(Color.cyan) ) {
        board.d3.setBackground(Color.lightGray);
    } else if ( board.d4.getBackground().equals(Color.cyan) ) {
        board.d4.setBackground(Color.lightGray);
    }
}

public static void check() {

    if ( correct = true ) {
        score++;
        board.l2.setText(Long.toString(score));
    }
    switch ( (int) score ) {
        case 50:
            TA.timer.setDelay(1750);
            TA.timer.setInitialDelay(1750);
            TA.timer.restart();
            break;
        case 100:
            TA.timer.setDelay(1450);
```

```
    TA.timer.setInitialDelay(1450);
    TA.timer.restart();
    break;
case 150:
    TA.timer.setDelay(1150);
    TA.timer.setInitialDelay(1150);
    TA.timer.restart();
    break;
case 200:
    TA.timer.setDelay(910);
    TA.timer.setInitialDelay(910);
    TA.timer.restart();
    break;
case 250:
    TA.timer.setDelay(790);
    TA.timer.setInitialDelay(790);
    TA.timer.restart();
    break;
case 300:
    TA.timer.setDelay(744);
    TA.timer.setInitialDelay(744);
    TA.timer.restart();
    break;
case 350:
    TA.timer.setDelay(700);
    TA.timer.setInitialDelay(700);
```

```
            TA.timer.restart();
            break;
        case 400:
            TA.timer.setDelay(650);
            TA.timer.setInitialDelay(650);
            TA.timer.restart();
            break;
        default:
            break;
    }
}

}
```

```
import java.awt.Color;
import java.awt.Container;
import java.awt.Cursor;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;

public class TA extends JFrame {

    private static Main_TA main = new Main_TA();

    private static TA sp = new TA();

    TA() {
        initComponents();
    }

    public static Timer timer = new Timer(2000, new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            Main_TA.set_after();
        }
    });

    void z1ActionPerformed(ActionEvent e) {
```

```
    if ( a1.getBackground().equals(Color.black) ) {
        a1.setBackground(Color.cyan);
        Main_TA.check();
    } else if ( b1.getBackground().equals(Color.black) ) {
        b1.setBackground(Color.cyan);
        Main_TA.check();
    } else if ( c1.getBackground().equals(Color.black) ) {
        c1.setBackground(Color.cyan);
        Main_TA.check();
    } else if ( d1.getBackground().equals(Color.black) ) {
        d1.setBackground(Color.cyan);
        Main_TA.check();
    }
}

void z2ActionPerformed(ActionEvent e) {
    if ( a2.getBackground().equals(Color.black) ) {
        a2.setBackground(Color.cyan);
        Main_TA.check();
    } else if ( b2.getBackground().equals(Color.black) ) {
        b2.setBackground(Color.cyan);
        Main_TA.check();
    } else if ( c2.getBackground().equals(Color.black) ) {
        c2.setBackground(Color.cyan);
        Main_TA.check();
    } else if ( d2.getBackground().equals(Color.black) ) {
```

```
        d2.setBackground(Color.cyan);
        Main_TA.check();
    }
}

void z3ActionPerformed(ActionEvent e) {
    if ( a3.getBackground().equals(Color.black) ) {
        a3.setBackground(Color.cyan);
        Main_TA.check();
    } else if ( b3.getBackground().equals(Color.black) ) {
        b3.setBackground(Color.cyan);
        Main_TA.check();
    } else if ( c3.getBackground().equals(Color.black) ) {
        c3.setBackground(Color.cyan);
        Main_TA.check();
    } else if ( d3.getBackground().equals(Color.black) ) {
        d3.setBackground(Color.cyan);
        Main_TA.check();
    }
}

void z4ActionPerformed(ActionEvent e) {
    if ( a4.getBackground().equals(Color.black) ) {
        a4.setBackground(Color.cyan);
        Main_TA.check();
    } else if ( b4.getBackground().equals(Color.black) ) {
```

```
        b4.setBackground(Color.cyan);
        Main_TA.check();
    } else if ( c4.getBackground().equals(Color.black) ) {
        c4.setBackground(Color.cyan);
        Main_TA.check();
    } else if ( d4.getBackground().equals(Color.black) ) {
        d4.setBackground(Color.cyan);
        Main_TA.check();
    }
}

void initComponents() {
    d1 = new JButton();
    d2 = new JButton();
    d3 = new JButton();
    d4 = new JButton();
    c1 = new JButton();
    c2 = new JButton();
    c3 = new JButton();
    c4 = new JButton();
    b1 = new JButton();
    b2 = new JButton();
    b3 = new JButton();
    b4 = new JButton();
    a1 = new JButton();
    a2 = new JButton();
```

```
        a3 = new JButton();
        a4 = new JButton();
        z1 = new JButton();
        z2 = new JButton();
        z3 = new JButton();
        z4 = new JButton();
        l1 = new JLabel();
        l2 = new JLabel();


setDefaultCloseOperation(javax.swing.WindowConstants.DO_NOTHING_
ON_CLOSE);
        setCursor(Cursor.HAND_CURSOR);
        setAlwaysOnTop(true);
        setResizable(false);

        setTitle("Tiny Army");
        Container contentPane = getContentPane();

        l1.setText("Your Scores are : ");

        z1.setText(" - O - ");
        z1.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                z1ActionPerformed(e);
```

```
    }
});
z2.setText(" - O - ");
z2.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        z2ActionPerformed(e);
    }
});
z3.setText(" - O - ");
z3.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        z3ActionPerformed(e);
    }
});
z4.setText(" - O - ");
z4.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        z4ActionPerformed(e);
    }
});

d1.setBackground(Color.white);
d2.setBackground(Color.white);
```

```
d3.setBackground(Color.white);
d4.setBackground(Color.white);
c1.setBackground(Color.white);
c2.setBackground(Color.white);
c3.setBackground(Color.white);
c4.setBackground(Color.white);
b1.setBackground(Color.white);
b2.setBackground(Color.white);
b3.setBackground(Color.white);
a1.setBackground(Color.white);
a2.setBackground(Color.white);
a3.setBackground(Color.white);
a4.setBackground(Color.white);

GroupLayout contentPaneLayout = new GroupLayout(contentPane);
contentPane.setLayout(contentPaneLayout);
contentPaneLayout.setHorizontalGroup(
        contentPaneLayout.createParallelGroup()
                .addGroup(contentPaneLayout.createSequentialGroup()
                        .addContainerGap()
                        .addGroup(contentPaneLayout.createParallelGroup()

.addGroup(contentPaneLayout.createSequentialGroup()
                                .addComponent(d1,
GroupLayout.PREFERRED_SIZE, 85, GroupLayout.PREFERRED_SIZE)
                                .addComponent(d2,
GroupLayout.PREFERRED_SIZE, 85, GroupLayout.PREFERRED_SIZE)
```

```
                                .addComponent(d3,
GroupLayout.PREFERRED_SIZE, 85, GroupLayout.PREFERRED_SIZE)

                                .addComponent(d4,
GroupLayout.PREFERRED_SIZE, 85, GroupLayout.PREFERRED_SIZE))

.addGroup(contentPaneLayout.createSequentialGroup()

                                .addComponent(c1,
GroupLayout.PREFERRED_SIZE, 85, GroupLayout.PREFERRED_SIZE)

                                .addComponent(c2,
GroupLayout.PREFERRED_SIZE, 85, GroupLayout.PREFERRED_SIZE)

                                .addComponent(c3,
GroupLayout.PREFERRED_SIZE, 85, GroupLayout.PREFERRED_SIZE)

                                .addComponent(c4,
GroupLayout.PREFERRED_SIZE, 85, GroupLayout.PREFERRED_SIZE))

.addGroup(contentPaneLayout.createSequentialGroup()

                                .addComponent(b1,
GroupLayout.PREFERRED_SIZE, 85, GroupLayout.PREFERRED_SIZE)

                                .addComponent(b2,
GroupLayout.PREFERRED_SIZE, 85, GroupLayout.PREFERRED_SIZE)

                                .addComponent(b3,
GroupLayout.PREFERRED_SIZE, 85, GroupLayout.PREFERRED_SIZE)

                                .addComponent(b4,
GroupLayout.PREFERRED_SIZE, 85, GroupLayout.PREFERRED_SIZE))

.addGroup(contentPaneLayout.createSequentialGroup()

                                .addComponent(a1,
GroupLayout.PREFERRED_SIZE, 85, GroupLayout.PREFERRED_SIZE)

                                .addComponent(a2,
GroupLayout.PREFERRED_SIZE, 85, GroupLayout.PREFERRED_SIZE)

                                .addComponent(a3,
GroupLayout.PREFERRED_SIZE, 85, GroupLayout.PREFERRED_SIZE)
```

```
                                        .addComponent(a4,
GroupLayout.PREFERRED_SIZE, 85, GroupLayout.PREFERRED_SIZE))

.addGroup(contentPaneLayout.createSequentialGroup()

                                        .addComponent(z1,
GroupLayout.PREFERRED_SIZE, 85, GroupLayout.PREFERRED_SIZE)

                                        .addComponent(z2,
GroupLayout.PREFERRED_SIZE, 85, GroupLayout.PREFERRED_SIZE)

                                        .addComponent(z3,
GroupLayout.PREFERRED_SIZE, 85, GroupLayout.PREFERRED_SIZE)

                                        .addComponent(z4,
GroupLayout.PREFERRED_SIZE, 85, GroupLayout.PREFERRED_SIZE))

.addGroup(contentPaneLayout.createSequentialGroup()

                                        .addComponent(l1,
GroupLayout.PREFERRED_SIZE, 150, GroupLayout.PREFERRED_SIZE)

                                        .addComponent(l2,
GroupLayout.PREFERRED_SIZE, 30, GroupLayout.PREFERRED_SIZE)))

                                .addContainerGap(11, Short.MAX_VALUE))

        );

        contentPaneLayout.setVerticalGroup(

                contentPaneLayout.createParallelGroup()

                        .addGroup(contentPaneLayout.createSequentialGroup()

                                .addContainerGap()

.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)

                                        .addComponent(d1,
GroupLayout.PREFERRED_SIZE, 80, GroupLayout.PREFERRED_SIZE)
```

```
                    .addComponent(d2,
GroupLayout.PREFERRED_SIZE, 80, GroupLayout.PREFERRED_SIZE)

                    .addComponent(d3,
GroupLayout.PREFERRED_SIZE, 80, GroupLayout.PREFERRED_SIZE)

                    .addComponent(d4,
GroupLayout.PREFERRED_SIZE, 80, GroupLayout.PREFERRED_SIZE))

.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)

                    .addComponent(c1,
GroupLayout.PREFERRED_SIZE, 80, GroupLayout.PREFERRED_SIZE)

                    .addComponent(c2,
GroupLayout.PREFERRED_SIZE, 80, GroupLayout.PREFERRED_SIZE)

                    .addComponent(c3,
GroupLayout.PREFERRED_SIZE, 80, GroupLayout.PREFERRED_SIZE)

                    .addComponent(c4,
GroupLayout.PREFERRED_SIZE, 80, GroupLayout.PREFERRED_SIZE))

.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)

                    .addComponent(b1,
GroupLayout.PREFERRED_SIZE, 80, GroupLayout.PREFERRED_SIZE)

                    .addComponent(b2,
GroupLayout.PREFERRED_SIZE, 80, GroupLayout.PREFERRED_SIZE)

                    .addComponent(b3,
GroupLayout.PREFERRED_SIZE, 80, GroupLayout.PREFERRED_SIZE)

                    .addComponent(b4,
GroupLayout.PREFERRED_SIZE, 80, GroupLayout.PREFERRED_SIZE))

.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)
```

```
                    .addComponent(a1,
GroupLayout.PREFERRED_SIZE, 80, GroupLayout.PREFERRED_SIZE)

                    .addComponent(a2,
GroupLayout.PREFERRED_SIZE, 80, GroupLayout.PREFERRED_SIZE)

                    .addComponent(a3,
GroupLayout.PREFERRED_SIZE, 80, GroupLayout.PREFERRED_SIZE)

                    .addComponent(a4,
GroupLayout.PREFERRED_SIZE, 80, GroupLayout.PREFERRED_SIZE))

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)

.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)

                    .addComponent(z1,
GroupLayout.PREFERRED_SIZE, 95, GroupLayout.PREFERRED_SIZE)

                    .addComponent(z2,
GroupLayout.PREFERRED_SIZE, 95, GroupLayout.PREFERRED_SIZE)

                    .addComponent(z3,
GroupLayout.PREFERRED_SIZE, 95, GroupLayout.PREFERRED_SIZE)

                    .addComponent(z4,
GroupLayout.PREFERRED_SIZE, 95, GroupLayout.PREFERRED_SIZE))

.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)

                    .addComponent(l1,
GroupLayout.PREFERRED_SIZE, 40, GroupLayout.PREFERRED_SIZE)

                    .addComponent(l2,
GroupLayout.PREFERRED_SIZE, 40, GroupLayout.PREFERRED_SIZE))

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)
```

```
                        .addContainerGap(15, Short.MAX_VALUE))
        );
        pack();
        setLocationRelativeTo(getOwner());
    }

    JButton d1, d2, d3, d4;

    JButton c1, c2, c3, c4;

    JButton b1, b2, b3, b4;

    JButton a1, a2, a3, a4;

    JButton z1, z2, z3, z4;

    JLabel l1, l2;
}


import java.awt.Container;
import java.awt.Cursor;
import javax.swing.*;

public class TA_Scores extends JFrame {
```

```
    TA_Scores() {
        initComponents();
    }

    void initComponents() {
        l1 = new JLabel();
        l2 = new JLabel();
        l3 = new JLabel();
        t1 = new JLabel();
        t2 = new JLabel();


setDefaultCloseOperation(javax.swing.WindowConstants.DO_NOTHING_
ON_CLOSE);
        setCursor(Cursor.HAND_CURSOR);
        setResizable(false);

        setTitle("Tiny Army Scores");
        Container contentPane = getContentPane();

        l2.setText("Your Scores are : ");
        l1.setText("The Enemy has attacked you.");
        l3.setText(Main_TA.l3);
        t1.setText(Main_TA.t1);
        t2.setText(Main_TA.t2);
```

```
        GroupLayout contentPaneLayout = new GroupLayout(contentPane);
        contentPane.setLayout(contentPaneLayout);
        contentPaneLayout.setHorizontalGroup(
                contentPaneLayout.createParallelGroup()
                        .addGroup(contentPaneLayout.createSequentialGroup()
                                .addContainerGap()
                                .addGroup(contentPaneLayout.createParallelGroup()

.addGroup(contentPaneLayout.createSequentialGroup()
                                        .addGap(21)
                                        .addComponent(l1,
GroupLayout.PREFERRED_SIZE, 230, GroupLayout.PREFERRED_SIZE)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED))
                                        .addGap(7)

.addGroup(contentPaneLayout.createSequentialGroup()
                                        .addComponent(l2,
GroupLayout.PREFERRED_SIZE, 105, GroupLayout.PREFERRED_SIZE)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)
                                        .addGap(10)
                                        .addComponent(l3,
GroupLayout.PREFERRED_SIZE, 55, GroupLayout.PREFERRED_SIZE)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED))

.addGroup(contentPaneLayout.createSequentialGroup()
```

```
                                .addComponent(t1,
GroupLayout.PREFERRED_SIZE, 255, GroupLayout.PREFERRED_SIZE)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)

                                .addGap(10)

                                .addComponent(t2,
GroupLayout.PREFERRED_SIZE, 55, GroupLayout.PREFERRED_SIZE)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)))

                        .addContainerGap(11, Short.MAX_VALUE))

        );

        contentPaneLayout.setVerticalGroup(

                contentPaneLayout.createParallelGroup()

                        .addGroup(contentPaneLayout.createSequentialGroup()

                        .addContainerGap()

.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)

                                .addGap(21)

                                .addComponent(l1,
GroupLayout.PREFERRED_SIZE, 138, GroupLayout.PREFERRED_SIZE))

                        .addGap(7)

.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)

                                .addComponent(l2,
GroupLayout.PREFERRED_SIZE, 138, GroupLayout.PREFERRED_SIZE)

                                .addGap(10)

                                .addComponent(l3,
GroupLayout.PREFERRED_SIZE, 138, GroupLayout.PREFERRED_SIZE))
```

```
.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)
                                .addComponent(t1,
GroupLayout.PREFERRED_SIZE, 138, GroupLayout.PREFERRED_SIZE)
                                .addGap(10)
                                .addComponent(t2,
GroupLayout.PREFERRED_SIZE, 138, GroupLayout.PREFERRED_SIZE))
                            .addContainerGap(15, Short.MAX_VALUE))
        );
        pack();
        setLocationRelativeTo(getOwner());
    }

    JLabel l1, l2, l3, t1, t2;

}
```

```
import java.awt.Container;
import javax.swing.*;

public class Exit extends JFrame {

    Exit() {
        initComponents();
    }

    void initComponents() {
        ex1 = new JLabel();
        ex2 = new JLabel();


setDefaultCloseOperation(javax.swing.WindowConstants.DO_NOTHING_
ON_CLOSE);
        setUndecorated(true);
        setResizable(false);

        setTitle("Thanks");
        Container contentPane = getContentPane();
        ex1.setText("Thanks For Playing ! ! !");
        ex2.setText("Have a nice day ahead ! ! !");
```

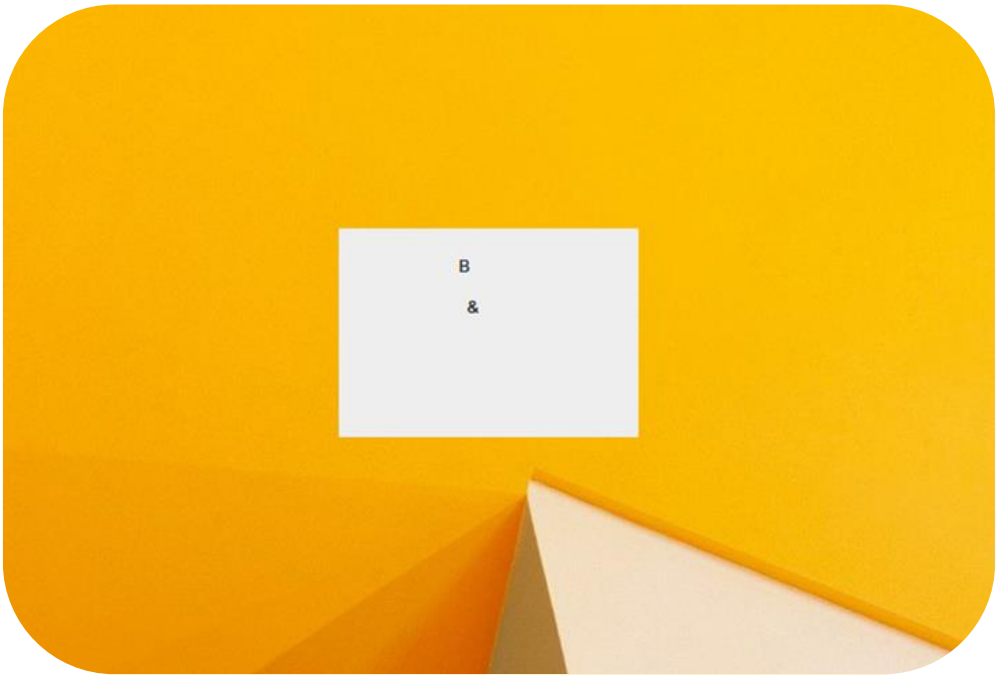
```
        GroupLayout contentPaneLayout = new GroupLayout(contentPane);
        contentPane.setLayout(contentPaneLayout);
        contentPaneLayout.setHorizontalGroup(
                contentPaneLayout.createParallelGroup()
                        .addGroup(contentPaneLayout.createSequentialGroup()
                                .addContainerGap()
                                .addGroup(contentPaneLayout.createParallelGroup()

.addGroup(contentPaneLayout.createSequentialGroup()
                                        .addGap(26)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)
                                        .addComponent(ex1,
GroupLayout.PREFERRED_SIZE, 130, GroupLayout.PREFERRED_SIZE)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED))

.addGroup(contentPaneLayout.createSequentialGroup()
                                        .addGap(26)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)
                                        .addComponent(ex2,
GroupLayout.PREFERRED_SIZE, 170, GroupLayout.PREFERRED_SIZE)

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)))
                                .addContainerGap(11, Short.MAX_VALUE))
        );
        contentPaneLayout.setVerticalGroup(
                contentPaneLayout.createParallelGroup()
```

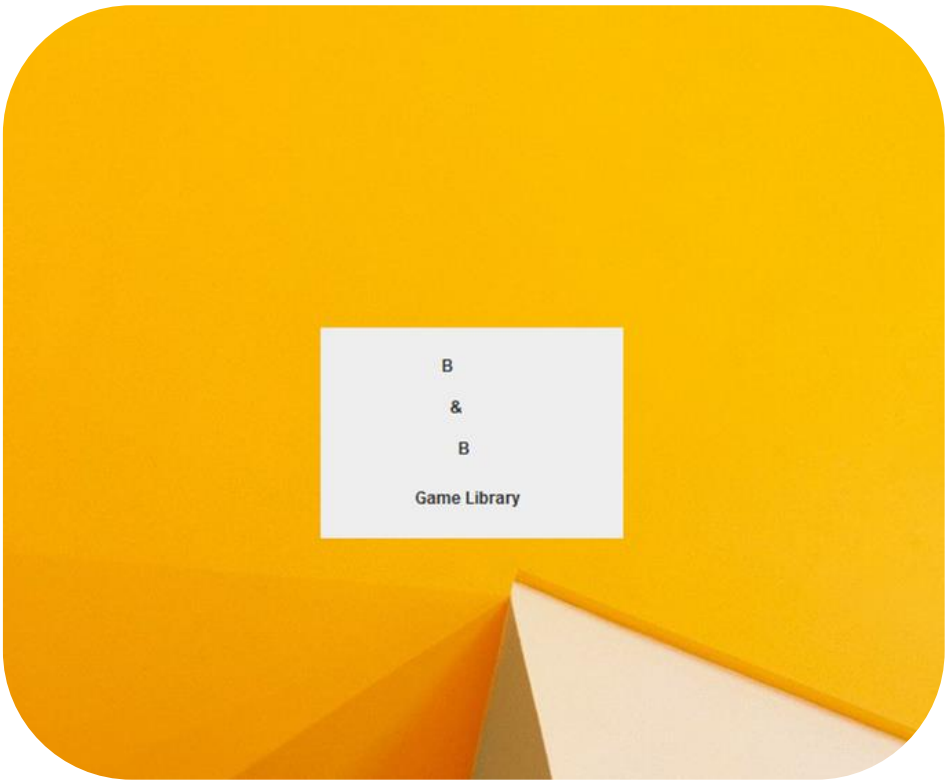
```
                    .addGroup(contentPaneLayout.createSequentialGroup()
                            .addContainerGap()

.addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)
                            .addGap(63)

.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)
                                    .addComponent(ex1,
GroupLayout.PREFERRED_SIZE, 30, GroupLayout.PREFERRED_SIZE))
                            .addGap(23)

.addGroup(contentPaneLayout.createParallelGroup(GroupLayout.Alignm
ent.BASELINE)
                                    .addComponent(ex2,
GroupLayout.PREFERRED_SIZE, 40, GroupLayout.PREFERRED_SIZE))
                            .addGap(43)
                            .addContainerGap(15, Short.MAX_VALUE))
        );
        pack();
        setLocationRelativeTo(getOwner());
    }

    JLabel ex1, ex2;

}
```

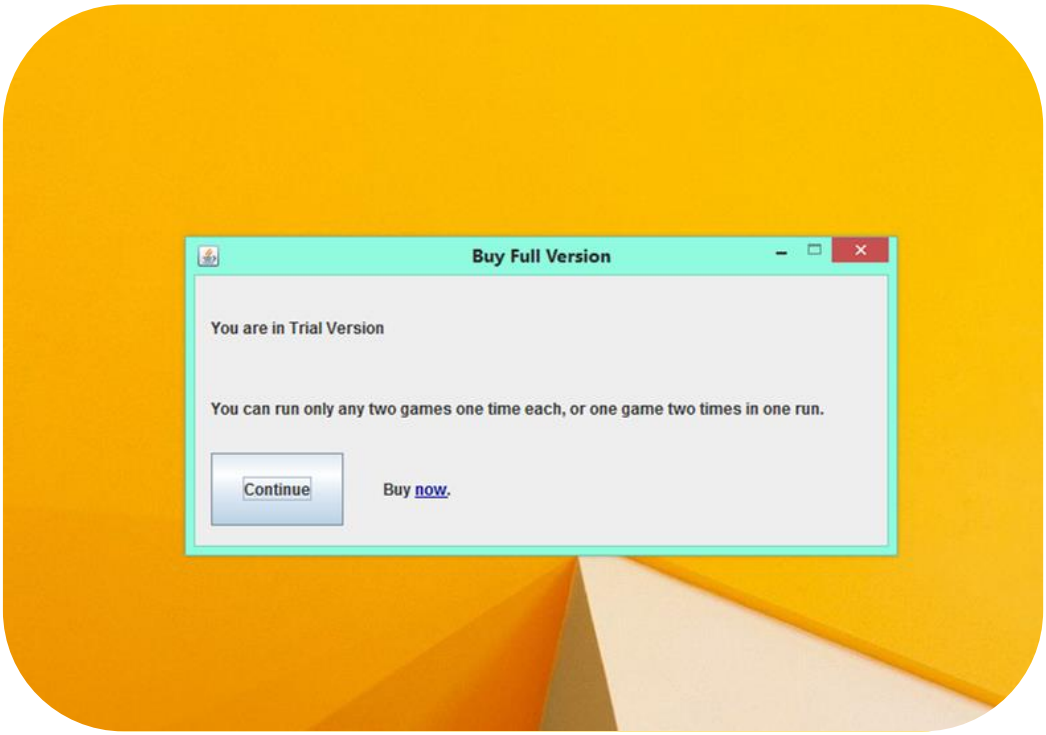
Output

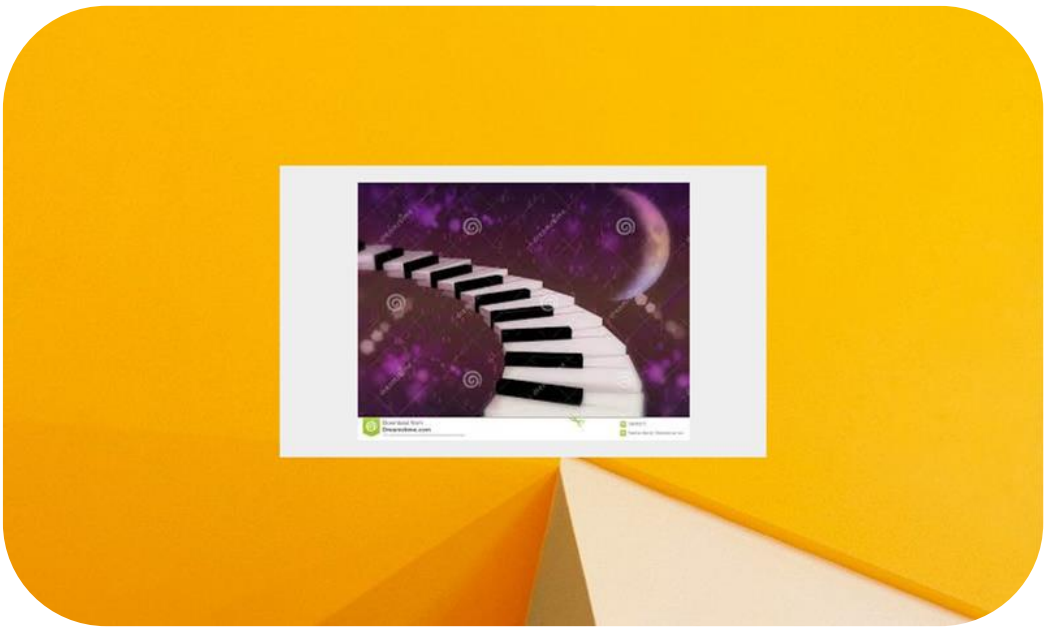
Bibliography

- CORDOVA Computer Applications Class X
- www.google.com
- www.quora.com
- netbeans.org
- www.stackoverflow.com
- www.oracle.com
- www.tutorialspoint.com
- www.geeksforgeeks.org
- tips4java.wordpress.com
- www.codeproject.com
- www.javatpoint.com
- www.javawithus.com
- beginnersbook.com
- www.java2us.com


B
&

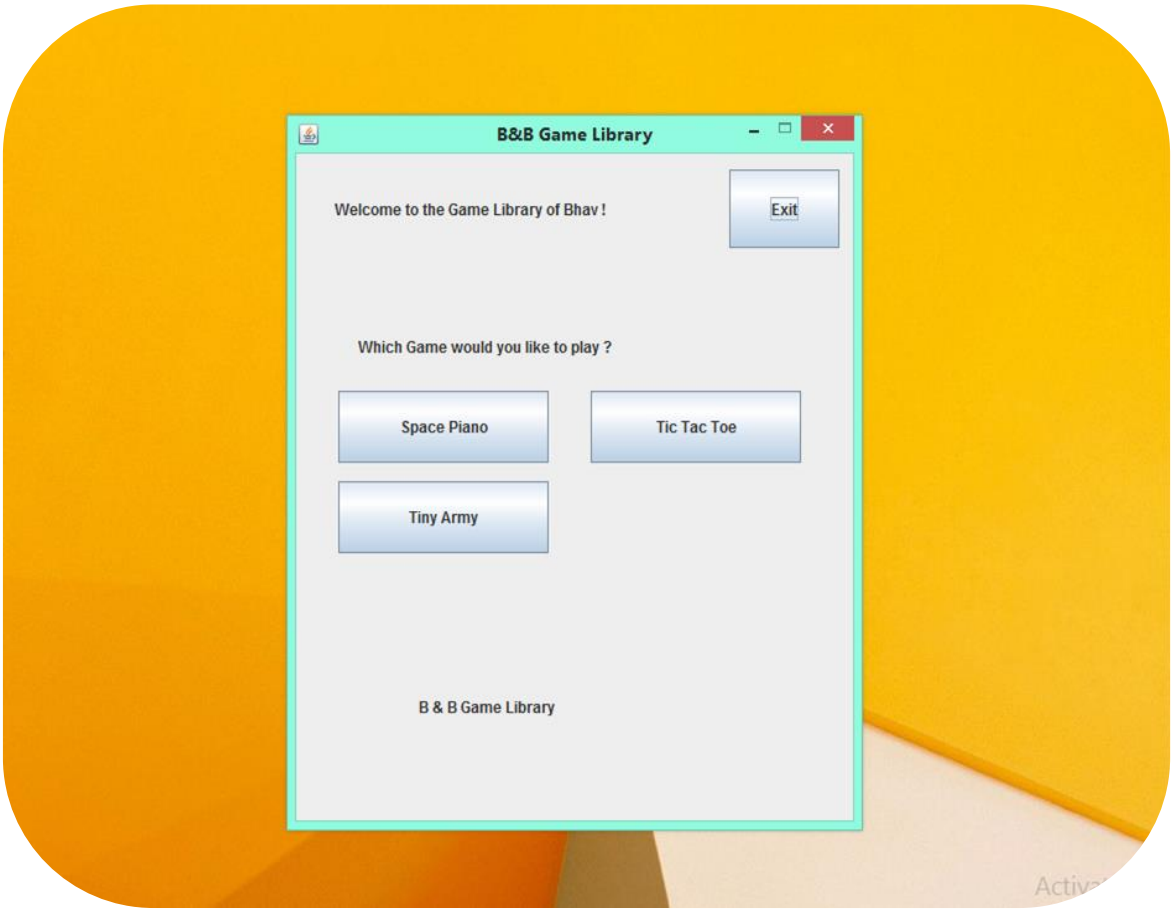

B
&
B
Game Library

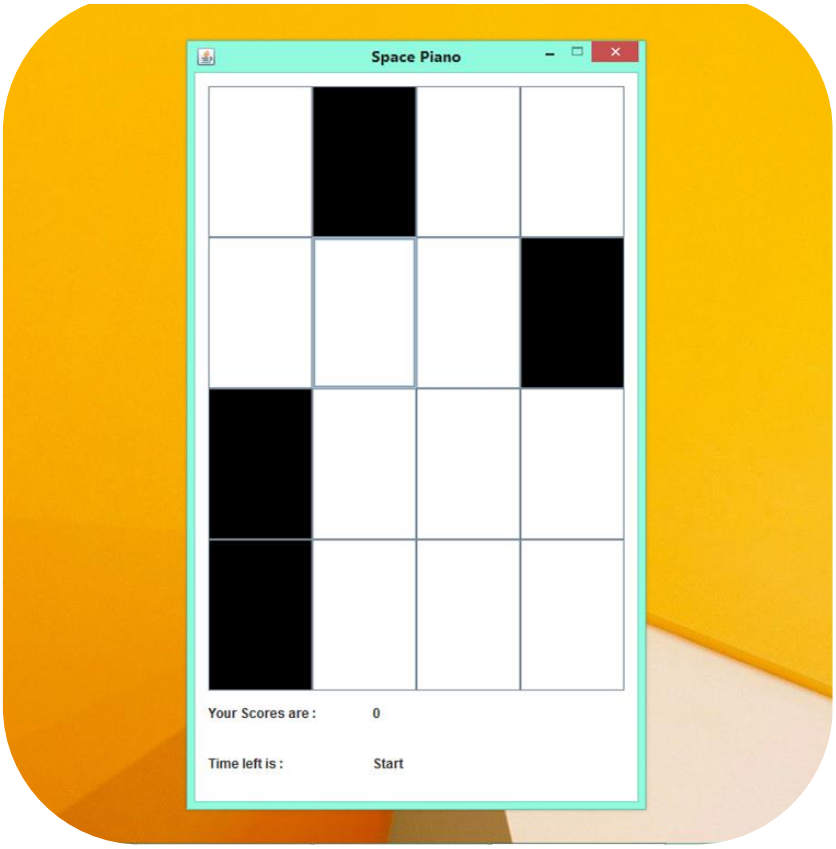

Buy Full Version
You are in Trial Version
You can run only any two games one time each, or one game two times in one run.
Continue
Buy now.

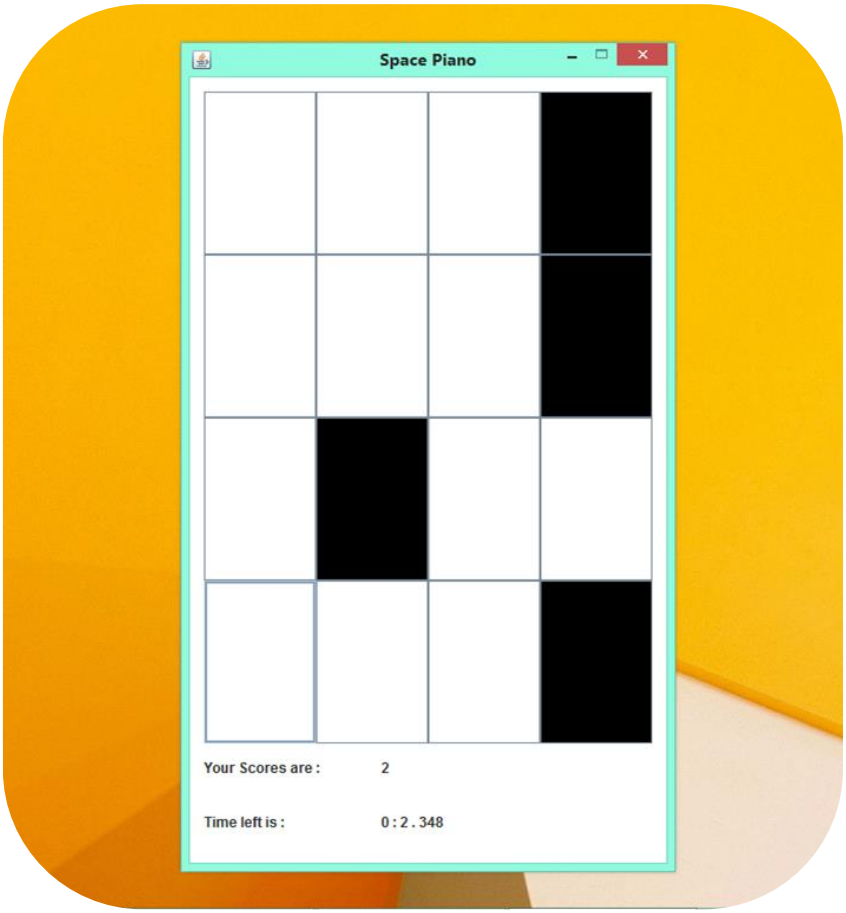


TIC TAC TOE

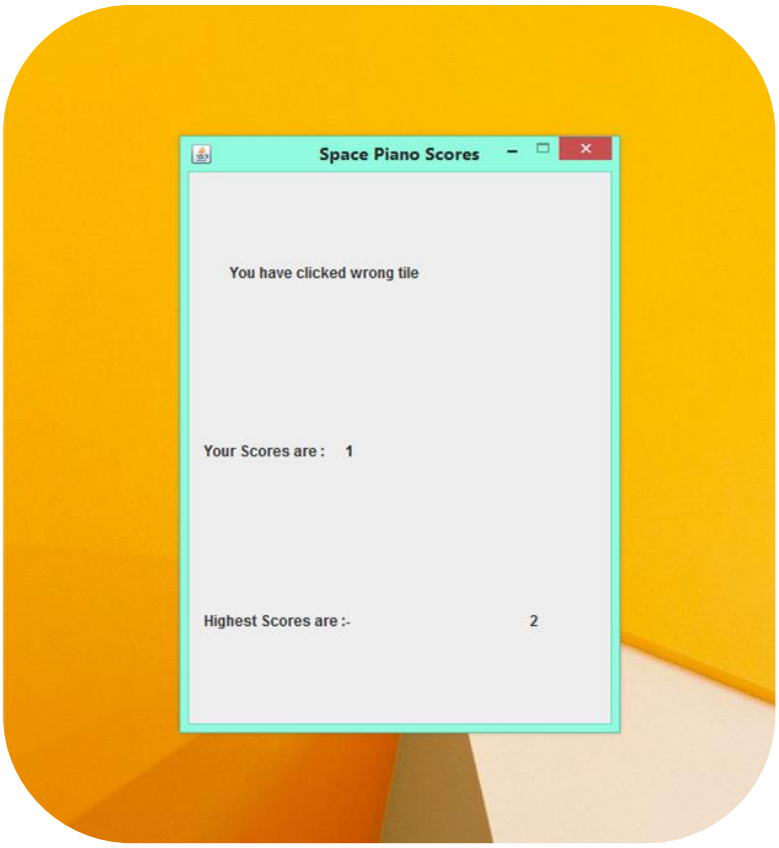
TINY ARMIES


B&B Game Library
Welcome to the Game Library of Bhav !
Exit
Which Game would you like to play ?
Space Piano
Tic Tac Toe
Tiny Army
B & B Game Library


Space Piano
Your Scores are : 0
Time left is : Start

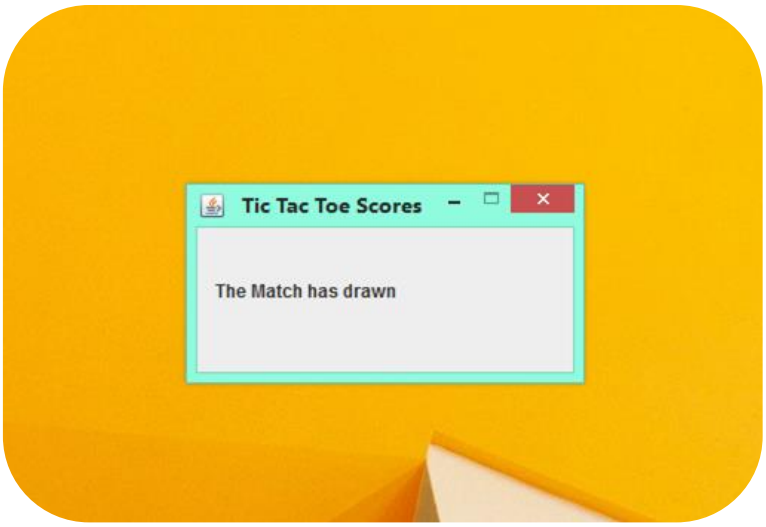

Space Piano
Your Scores are : 2
Time left is : 0 : 2 . 348

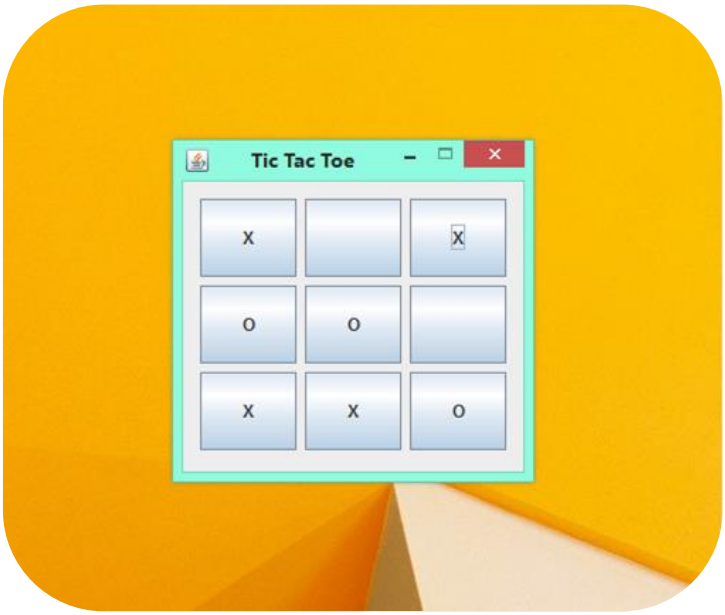
Space Piano Scores
Time Out
Your Scores are : 1
Highest Scores are :- 2

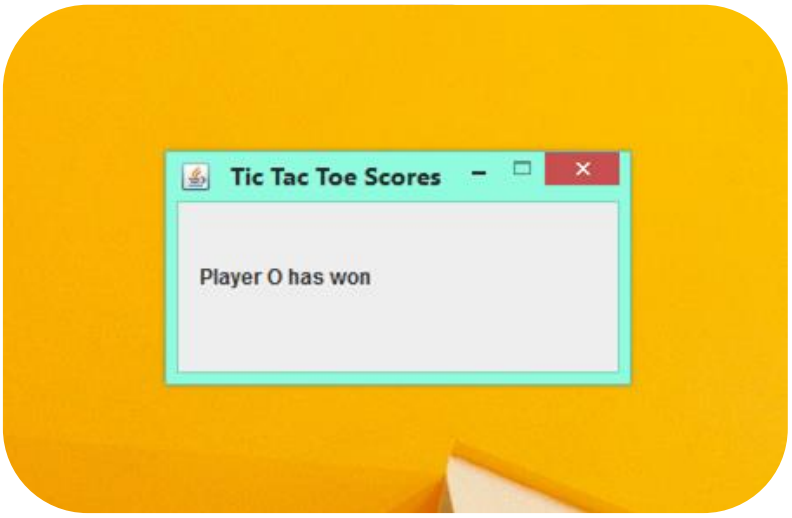

Space Piano Scores
You have clicked wrong tile
Your Scores are : 1
Highest Scores are :- 2

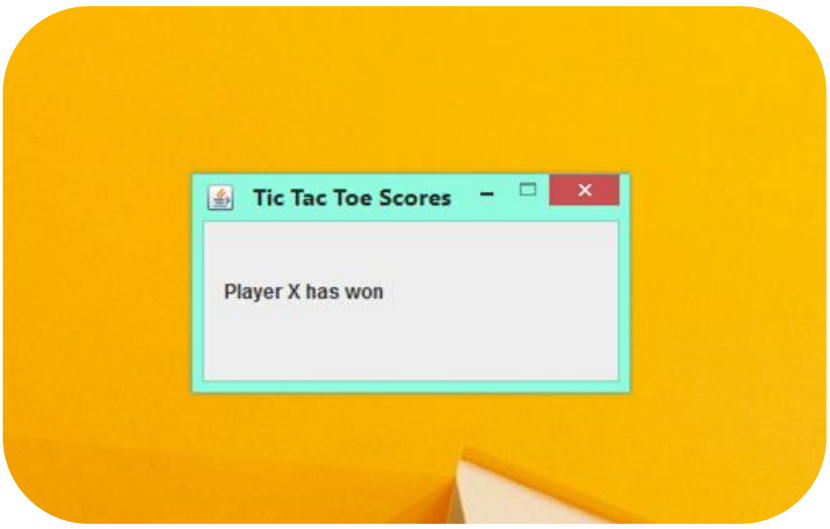
Tic Tac Toe

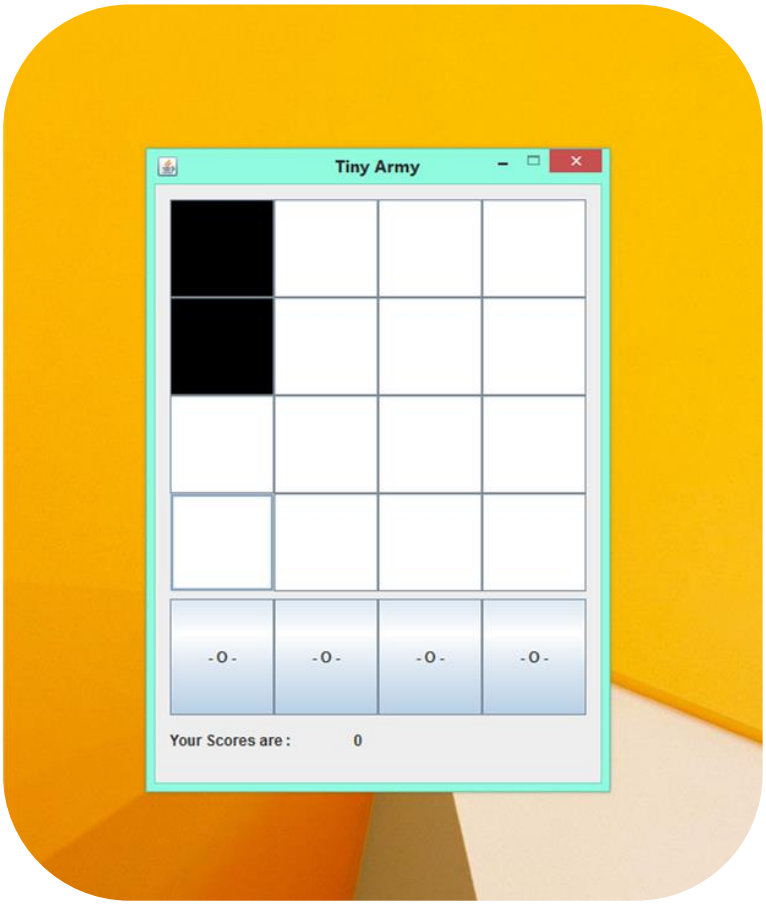
Tic Tac Toe
X
O
O
X
X
X
O
O

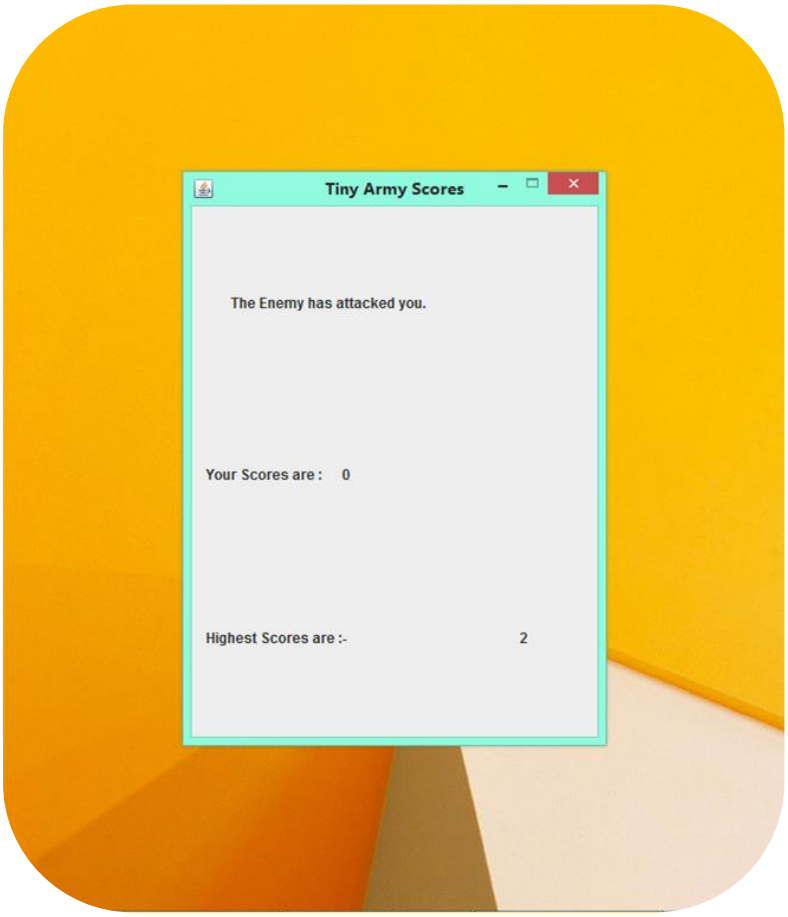

Tic Tac Toe Scores
The Match has drawn

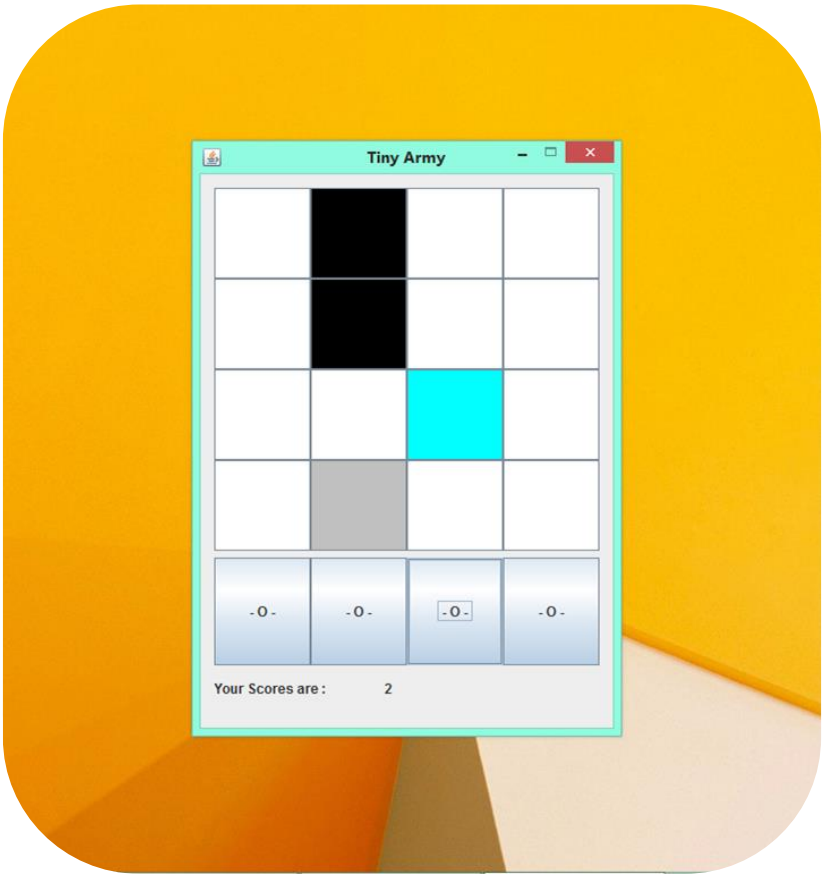

Tic Tac Toe
X
X
O
O
X
X
O

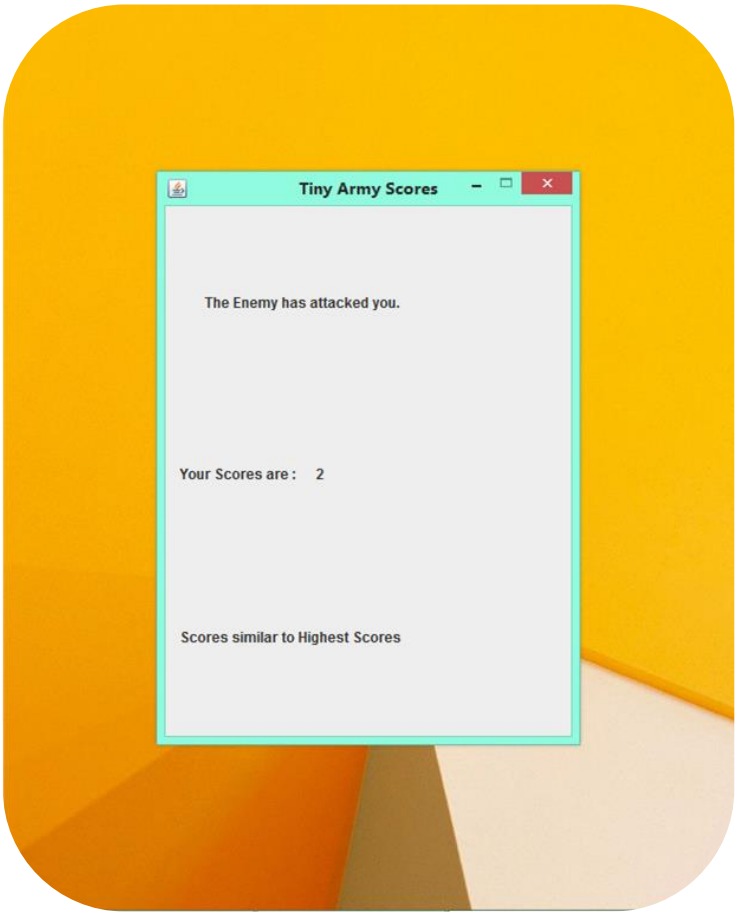

Tic Tac Toe Scores
Player O has won

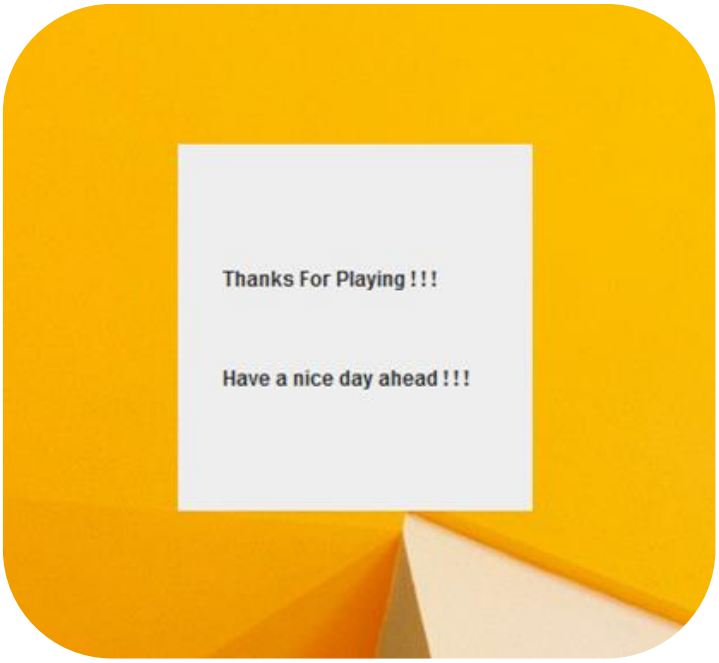

Tic Tac Toe Scores
Player X has won


Tiny Army
- O -
- O -
- O -
- O -
Your Scores are :
0


Tiny Army Scores
The Enemy has attacked you.
Your Scores are : 0
Highest Scores are :- 2


Tiny Army
- O -
- O -
- O -
- O -
Your Scores are :
2


Tiny Army Scores
The Enemy has attacked you.
Your Scores are : 2
Scores similar to Highest Scores


Thanks For Playing !!!
Have a nice day ahead !!!

Bibliography

- CORDOVA Computer Applications Class X
- www.google.com
- www.quora.com
- netbeans.org
- www.stackoverflow.com
- www.oracle.com
- www.tutorialspoint.com
- www.geeksforgeeks.org
- tips4java.wordpress.com
- www.codeproject.com
- www.javatpoint.com
- www.javawithus.com
- beginnersbook.com
- www.java2us.com