**SUMMER TRAINING REPORT**

**ON**

**‘OTHELLO MASTER’**

**Submitted in the partial fulfillment of the requirements for the award of degree**

**BACHELOR OF TECHNOLOGY**

**IN**

**INFORMATION TECHNOLOGY**

***Submitted By:***

***Student Name: VIDUSHI GARG***

***Roll No.: 07914803116***

***Semester: 5th***

***Batch: 2020***



**Maharaja Agrasen Institute of Technology, PSP area, Sector – 22, Rohini, New Delhi – 110085**

**GGSIP University**

**ACKNOWLEDGEMENT**

The satisfaction that accompanies that the successful completion of any task would be incomplete without the mention of the people whose ceaseless cooperation made it possible, whose constant guidance and encouragement crown all efforts with success.

I am grateful to my mentor **Mr. Sumeet Malik** for the guidance, inspiration and constructive suggestion that helped me in the preparation of this report.

I would like to thank all my friends for helping me in all measures of life and for their guidance, cooperation and moral support.

I would like to express my gratitude to all those who directly or indirectly helped me throughout the training for the successful completion.

Last but not the least; I would like to thanks all the staff at **Pepcoding** for being so helpful during this summer training.

**Vidushi Garg**

**B.Tech (Information Technology | 3rd year)**

**07914803116**

**MAIT, DELHI**

**CERTIFICATE**

****

**DECLARATION**

I hereby declare that the project entitled “**Othello Master**” submitted for “**Summer Training Report**” to Maharaja Agrasen Institute of Technology, is and record of an original work done by me under the guidance of Mr. Sumeet Malik, Pepcoding, and this project is submitted in the partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Information Technology.

**Vidushi Garg**

**B.Tech (Information Technology | 3rd year)**

**07914803116**

**MAIT, DELHI**

**PREFACE**

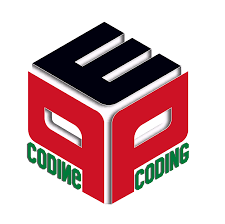
To develop a project in any language means to combine all the features of the language in a well-planned and systematic way so as to develop a useful application. This is the report for the project "Othello Master" which is a purely JAVA based application. In terms of its ultimate objectives, it gives an experience of a realistic application with entertainment right at your fingerprints.

The main aims of the project were set out as follows, with the majority of the project time to be spent on the second objective.

* Produce a computer program implementing the game of Othello.
* Create a Reversi game playing program where the discs on the board can only move accordingly to what is defined as a legal move, by the rules of the game.
* Include a user-friendly graphical interface. It was decided to program the project using Java.

Java classes allow for easy division of the project, providing a clear understanding of the required interfaces. We have tried a lot to achieve all the characteristics of good software. In later version it will be more refined and integrated.

**COMPANY PROFILE**

****

**PEPCODING**

**Pepcoding** Institute is an Institute dedicated to teaching Core Computer Technologies to students and professionals. Founded by **Sumeet Malik** in 2017, the institute has had the opportunity to teach thousands of students and professionals.

The teaching philosophy deployed, strives to create in-depth knowledge about the subject at hand. We believe that depth is an essential ingredient to achieve heights in software development. Students from our Institute have proven this point by their work in the industry, worldwide. Our Institute strives to provide a bigger picture to their students, the etiquette, the way of thinking which goes beyond normal technical jargon.

With an explosion of information on all software fronts, it is becoming increasingly difficult for students and professionals to manage learning. We help students achieve their long term technical goals by concentrating on core skills and technologies. We help them achieve proficiency by inculcating in them practical skills. We help them pace their learning, develop their rhythm and be fluent and graceful in their skills.

Coding in its most essential form is nothing but problem solving. It deals with figuring out logical solutions and implementing them optimally. The process of learning to code transforms a person into an analytical thinker with strong mathematical and logical skills.

Having been a standalone field for a long time, the software industry has now come neck to neck with most of the other industries by close coordination and easy integration. From the smallest light bulb to the most massive aircrafts, every industry is now implementing the use of computer programming for the convenience of the end user. A good command over coding can thus open a wide variety of employment opportunities.

Pepcoding strongly believe that with the right guidance and perfect determination, any student willing to learn programming can become a master of coding. All team members at Pepcoding are aces of their respective field and the share the highest level of commitment towards quality teaching and student success and satisfaction. The present structure of computer education in colleges and universities is not aligned to the needs of the IT Industry. Students have no place to go and bridge this huge gap.

**TABLE OF CONTENTS**

1. **CHAPTER 1**
   1. About the programming language- “Java”
   2. Java Architecture
   3. Java Platform
   4. Java Features
2. **CHAPTER 2**
   1. Java Environment
   2. JDK
   3. JRE
   4. JVM
   5. Software used
      1. Eclipse
   6. AWT package
   7. Swing package
3. **CHAPTER 3**
   1. Description
   2. Project Aims and Objectives
   3. Rules of Othello
   4. Implementation
4. **CHAPTER 4**

4.1. Output Screenshots

1. **CONCLUSION**
2. **BIBLIOGRAPHY**

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| S. No. | Description | Page Number |
| 1 | Java Architecture | 12 |
| 2 | Java Development Kit | 17 |
| 3 | Java Virtual Machine | 18 |
| 4 | Swing Package | 20 |
| 5 | Screenshots of code | 26-28 |
| 6 | Output Screenshots | 29-30 |

1. **CHAPTER 1**
   1. **ABOUT THE PROGRAMMIMG LANGUAGE “JAVA”**



With the invention of microprocessors, the world is scientifically developed with sophisticated equipments, systems, and devices. Microprocessors are used in computers, televisions, and fax machines. Even the hand-held devices such as pagers, PDAs (Personal Digital Assistant), and cell phones make use of microprocessors. All these electronic devices are helpful because of their communication capabilities. With the increasing capabilities and decreasing cost of information processing and networking technologies, the network is growing rapidly for transmitting information through electronic systems.

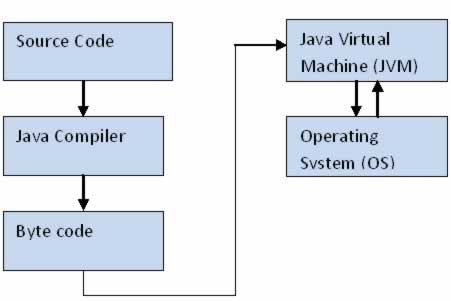
Internet is the network of networks between different types of computers located at different places to transmit information. Information can reach to any place in the world quickly at a cheaper rate through the Internet. Thus, the Internet has made the world a global village for information exchange. The emerging infrastructure of electronic devices and interconnected computer networks create an environment that presents new challenges to software industries. for this emerging computing environment, Java process to be a well – suited programming language. it is found suitable for networked environments involving a great variety of computer and devices.

Java has many characteristics that have contributed to its popularity:

* **Platform independence** - Many languages are compatible with only one platform. Java was specifically designed so that it would run on any computer, regardless if it was running Windows, Linux, Mac, Unix or any of the other operating systems.
* **Simple and easy to use** - Java's creators tried to design it so code could be written efficiently and easily.
* **Multi-functional** - Java can produce many applications from command-line programs to applets to Swing windows (basically, sophisticated graphical user interfaces).

Java does have some drawbacks. Since it has automated garbage collection, it can tend to use more memory than other similar languages. There are often implementation differences on different platforms, which have led to Java being described as a "write once, test everywhere" system. Lastly, since it uses an abstract "virtual machine", a generic Java program doesn't have access to the Native API's on a system directly. None of these issues are fatal, but it can mean that Java isn't an appropriate choice for a particular piece of software.

* 1. **Java Architecture**



Java combines both the approaches of compilation and interpretation. First, java compiler compiles the source code into bytecode. At the run time, Java Virtual Machine (JVM) interprets this bytecode and generates machine code which will be directly executed by the machine in which java program runs.

* 1. ooxWord://word/media/image222.binooxWord://word/media/image223.bin**The Java Platform**

One thing that distinguished Java from some other languages is its ability to run the same compiled code across multiple operating systems.In other languages, the source code (code that is written by the programmer), is compiled by a compiler into an executable file. This file is in machine language, and is intended for a single operating system/processor combination, so the programmer would have to re-compile the program seperately for each new operating system/processor combination.Java is different in that it does not compile the code directly into machine language code. Compilation creates bytecode out of the source code. Bytecode generally looks something like this:

a7 f4 73 5a 1b 92 7d

When the code is run by the user, it is processed by something called the Java Virtual Machine (JVM). The JVM is essentially an interpreter for the bytecode. It goes through the bytecode and runs it. There are different versions of the JVM that are compatible with each OS and can run the same code. There is virtually no difference for the end-user, but this makes it a lot easier for programmers doing software development.

* 1. **Java features**
* Simple

Java was designed to be easy for the professional programmer. It is easy to learn and can be used effectively. If you are an experienced C++ programmer, moving to Java will require very little effort.

* Secure

There is a concept of applets in Java which can be downloaded without fear or virus or malicious content, because the Java programs are confined to Java execution environment and are not allowed to access other parts of the CPU.

* Portable

The Java programs called Applets run in the JVM (Java virtual machine) environment that is in every browser therefore the programs can run anywhere.

* ObjectOriented

Java Classes follow the Oops concept of encapsulation, inheritance, and polymorphism.

* Robust

Garbage collection and Exception handling make Java a robust language. In garbage collection the user doesn’t have to bother about the memory allocation as, when the object is no longer in use it is automatically deleted to release memory space.

* Multithreaded

A single threaded application has one thread of execution running at all times and such programs can do only one task a time. A multi-threaded application can have several threads of execution running independently and simultaneously. These threads may communicate and cooperate and will appear to be a single program to the user.

* Interpreted

The Java code is compiled into the byte code, which is the class file. The byte code is then interpreted to the machine language by the JVM environment.

**Some Other Features of Java Programming**

* Encapsulation

Encapsulation is the mechanism that binds together code and the data it manipulates, and keeps both safe outside interference and misuse. It is a protective wrapper that prevents the code and data from being arbitrarily accessed by other code defined outside the wrapper. Encapsulation is the capability to represent, denote and handle information at a higher level that is inherent to a computer or base language. Variables and methods are formerly known as instance variables and instance methods to distinguish from class variables and class methods.

* Inheritance

Inheritance is the process by which one object acquires the properties of another object.

Classes inherit state and behavior from their superclass. A class is a blueprint or prototype that defines the variables and methods common to all objects of a certain kind.

Object oriented systems allow classes to be defined in terms of other classes. For example, mountain bikes, racing bikes and tandems are all subclasses of the bicycle class. Similarly, the bicycle class is the superclass of mountain bikes, racing bikes and tandems. Each subclass inherits state (in the form of variable declarations) from the superclass. Mountain bikes, racing bikes and tandems share some states: Cadence, speed, and the like. Also each subclass inherits methods from the superclass.

BenefitsofInheritance:

Subclasses provide specialized behaviors on the basis of common elements provided by the superclass. Through the use of inheritance programmers can reuse this code in the superclass many times.

* Abstraction

Abstraction, is this process of categorizing data. Consider that a person is looking for a frame in

an optician's shop. To be able to choose a frame from amongst the various types of frames

available, he has to first identify the attributes he is looking far. Once he has identified the

attributes, he has with him a category or class of frames. Similarly, to model any real life objects

in OOPS an "object" has to be instantiated from a specific "class". This basic process of forming

a class is known as **abstraction.**

1. **CHAPTER 2**
   1. **Java Environment**

Using Java, it is impossible to build applications that access libraries and applications in other languages; that is, unless you are willing to use the underlying operating system (OS). This is not a failure of Java, but simply a result of its lack of external device support. Until such support is provided, Java developers must find alternative methods to interact with libraries and applications written in other languages. One alternative is the Java Native Interface (JNI). These techniques provide a stop-gap measure until device support becomes part of the Java environment.

The JNI provides a documented and supported specification that allows programs written in other languages to be called from Java. Calling Java from an application written in another language is often referred to as *embedding*, and the process requires an understanding of the Invocation API. The JNI also provides a way for applications written in other languages to call Java.

**SETTING UP THE ENVIRONMENT IN JAVA**

Java applications are typically compiled to **bytecode** that can run on any Java virtual machine (JVM) regardless of computer architecture. The latest version is **Java 8**.

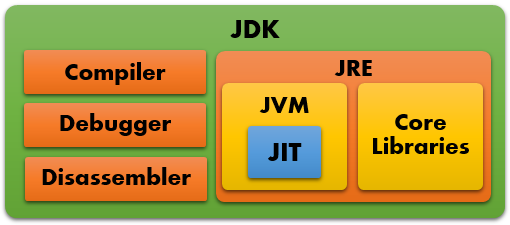
Below are the environment settings for both Linux and Windows. **JVM**, **JRE** and **JDK** all three are platform dependent because configuration of each Operating System is different. But, Java is platform independent.

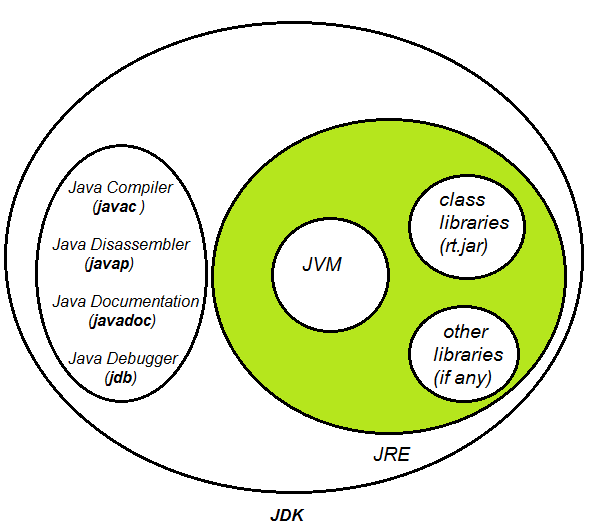
There are few things which must be clear before setting up the environment

* 1. **JDK**(Java Development Kit) : JDK is intended for software developers and includes development tools such as the Java compiler, Javadoc, Jar, and a debugger

.

* 1. **JRE**(Java Runtime Environment) : JRE contains the parts of the Java libraries required to run Java programs and is intended for end users. JRE can be view as a subset of JDK.
  2. **JVM**(Java Virtual Machine) : JVM is a component which provides an environment for running Java programs. JVM interprets the bytecode into machine code which will be executed the machine in which the Java program runs.

****

****

* 1. **SOFTWARE USED**
     1. **ECLIPSE**



Eclipse is an [integrated development environment](https://en.wikipedia.org/wiki/Integrated_development_environment) (IDE) used in [computer programming](https://en.wikipedia.org/wiki/Computer_programming), and is the most widely used Java IDE. It contains a base [workspace](https://en.wikipedia.org/wiki/Workspace) and an extensible [plug-in](https://en.wikipedia.org/wiki/Plug-in_(computing)) system for customizing the environment. Eclipse is written mostly in [Java](https://en.wikipedia.org/wiki/Java_(programming_language)) and its primary use is for developing Java applications, but it may also be used to develop applications in other [programminglanguages](https://en.wikipedia.org/wiki/Programming_language) viaplugins,including [Ada](https://en.wikipedia.org/wiki/Ada_(programming_language)), [ABAP](https://en.wikipedia.org/wiki/ABAP), [C](https://en.wikipedia.org/wiki/C_(programming_language)), [C++](https://en.wikipedia.org/wiki/C%2B%2B), [C#](https://en.wikipedia.org/wiki/C_Sharp_(programming_language)), [COBOL](https://en.wikipedia.org/wiki/COBOL), [D](https://en.wikipedia.org/wiki/D_(programming_language)) [Fortran](https://en.wikipedia.org/wiki/Fortran), [Haskell](https://en.wikipedia.org/wiki/Haskell_(programming_language)), [JavaScript](https://en.wikipedia.org/wiki/JavaScript), [Julia](https://en.wikipedia.org/wiki/Julia_(programming_language)), [Lasso](https://en.wikipedia.org/wiki/Lasso_(programming_language)), [Lua](https://en.wikipedia.org/wiki/Lua_(programming_language)),NATURAL, [Perl](https://en.wikipedia.org/wiki/Perl), [PHP](https://en.wikipedia.org/wiki/PHP), [Prolong](https://en.wikipedia.org/wiki/Prolog), [Python](https://en.wikipedia.org/wiki/Python_(programming_language)), [R](https://en.wikipedia.org/wiki/R_(programming_language)), [Ruby](https://en.wikipedia.org/wiki/Ruby_(programming_language)) (including [Ruby on Rails](https://en.wikipedia.org/wiki/Ruby_on_Rails) framework), [Rust](https://en.wikipedia.org/wiki/Rust_(programming_language)), [Scala](https://en.wikipedia.org/wiki/Scala_(programming_language)" \o "Scala (programming language)), [Clojure](https://en.wikipedia.org/wiki/Clojure" \o "Clojure), [Groovy](https://en.wikipedia.org/wiki/Groovy_(programming_language)), [Scheme](https://en.wikipedia.org/wiki/Scheme_(programming_language)), and [Erlang](https://en.wikipedia.org/wiki/Erlang_(programming_language)" \o "Erlang (programming language)). It can also be used to develop documents with [LaTeX](https://en.wikipedia.org/wiki/LaTeX" \o "LaTeX) (via a TeXlipse plug-in) and packages for the software [Mathematica](https://en.wikipedia.org/wiki/Mathematica" \o "Mathematica). Development environments include the Eclipse Java development tools (JDT) for Java and Scala, Eclipse CDT for C/C++, and Eclipse PDT for PHP, among others.

The Eclipse [software development kit](https://en.wikipedia.org/wiki/Software_development_kit) (SDK), which includes the Java development tools, is meant for Java developers. Users can extend its abilities by installing plug-ins written for the Eclipse Platform, such as development toolkits for other programming languages, and can write and contribute their own plug-in modules. Since the introduction of the [OSGi](https://en.wikipedia.org/wiki/OSGi" \o "OSGi) implementation ([Equinox](https://en.wikipedia.org/wiki/Equinox_(OSGi))) in version 3 of Eclipse, plug-ins can be plugged-stopped dynamically and are termed (OSGI) bundles.

Eclipse [software development kit](https://en.wikipedia.org/wiki/Software_development_kit) (SDK) is [free and open-source software](https://en.wikipedia.org/wiki/Free_and_open-source_software), released under the terms of the [Eclipse Public License](https://en.wikipedia.org/wiki/Eclipse_Public_License), although it is incompatible with the [GNU General Public License](https://en.wikipedia.org/wiki/GNU_General_Public_License). It was one of the first IDEs to run under [GNU Classpath](https://en.wikipedia.org/wiki/GNU_Classpath) and it runs without problems under [IcedTea](https://en.wikipedia.org/wiki/IcedTea" \o "IcedTea).

* 1. **AWT Package**

The Abstract Window Toolkit (AWT) package enables you to create GUIs in your applets and applications. Different Windows API from different platforms were looked at and components were identified that were common to all of them. The AWT API communicates with the platform's native API's as to give your application the native look and feel. Because they associate with their own native screen resources, AWT components are called heavyweight components.

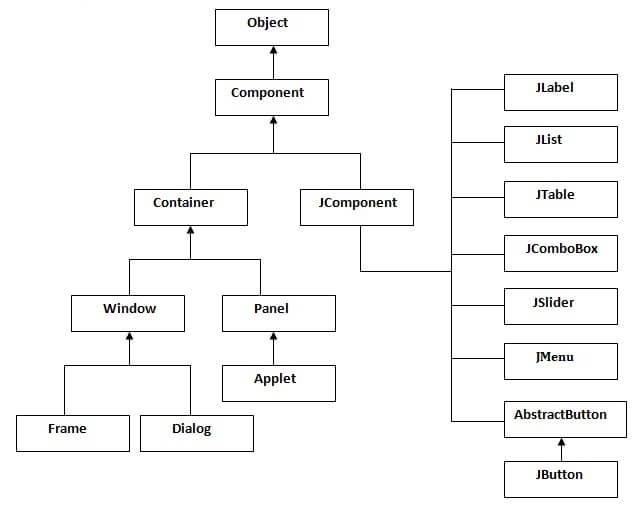
[Swing components](http://www.esus.com/javaindex/j2se/jdk1.2/javaxswing/javaxswing.html) do not associate with native resources and are called lightweight components. The AWT consists of *components*, both container and non-container ones (Button, Checkbox, Choice.) Container components (e.g. Frame, Dialog, Window, and Panel) control the layout of other components. As containers are themselves components, they can be placed inside other containers. Check out these links and subcategories on how to create and work with these containers and components.

**Subcategories**

|  |  |
| --- | --- |
| * [Button](http://www.esus.com/javaindex/j2se/jdk1.2/javaawt/awtbuttons/awtbuttons.html) * [Canvas](http://www.esus.com/javaindex/j2se/jdk1.2/javaawt/awtcanvas/awtcanvas.html) * [Checkbox and RadioButton](http://www.esus.com/javaindex/j2se/jdk1.2/javaawt/awtcheckradio/awtcheckradio.html) * [Choice](http://www.esus.com/javaindex/j2se/jdk1.2/javaawt/awtchoices/awtchoices.html) * [Colour](http://www.esus.com/javaindex/j2se/jdk1.2/javaawt/awtcolor/awtcolor.html) * [Data Transfer (java.awt.datatransfer.\*)](http://www.esus.com/javaindex/j2se/jdk1.2/javaawt/awtdatatransfer/awtdatatransfer.html) * [Dialog](http://www.esus.com/javaindex/j2se/jdk1.2/javaawt/awtdialog/awtdialog.html) * [Drag and Drop (java.awt.dnd.\*)](http://www.esus.com/javaindex/j2se/jdk1.2/javaawt/awtdragdrop/awtdragdrop.html) * [Event](http://www.esus.com/javaindex/j2se/jdk1.2/javaawt/awtevents/awtevents.html) * [Font](http://www.esus.com/javaindex/j2se/jdk1.2/javaawt/awtfont/awtfont.html) * [Frame](http://www.esus.com/javaindex/j2se/jdk1.2/javaawt/awtframe/awtframe.html) * [Graphics](http://www.esus.com/javaindex/j2se/jdk1.2/javaawt/awtgraphics/awtgraphics.html) | * [Insets](http://www.esus.com/javaindex/j2se/jdk1.2/javaawt/awtinsets/awtinsets.html) * [Input Method Framework](http://www.esus.com/javaindex/j2se/jdk1.2/javaawt/awtimf/awtimf.html) * [Label](http://www.esus.com/javaindex/j2se/jdk1.2/javaawt/awtlabels/awtlabels.html) * [Layout Managers](http://www.esus.com/javaindex/j2se/jdk1.2/javaawt/awtlayout/awtlayout.html) * [List](http://www.esus.com/javaindex/j2se/jdk1.2/javaawt/awtlists/awtlists.html) * [Menu, MenuItem](http://www.esus.com/javaindex/j2se/jdk1.2/javaawt/awtmenu/awtmenu.html) * [Robot](http://www.esus.com/javaindex/j2se/jdk1.2/javaawt/awtrobot/awtrobot.html) * [Scrollbar](http://www.esus.com/javaindex/j2se/jdk1.2/javaawt/awtscrollbar/awtscrollbar.html) * [ScrollPane](http://www.esus.com/javaindex/j2se/jdk1.2/javaawt/awtscrollpane/awtscrollpane.html) * [Textfield, TextArea](http://www.esus.com/javaindex/j2se/jdk1.2/javaawt/awttextfields/awttextfields.html) * [Toolkit](http://www.esus.com/javaindex/j2se/jdk1.2/javaawt/awttoolkit/awttoolkit.html) |

* 1. **SWING Package**

Swing is not an acronym. It packages a set of GUI components. Unlike AWT components, that are associated to native screen resources (heavyweight), Swing components draw themselves on the screen (lightweight). This result in slower execution but a Swing application will look the same on all platforms. Because Swing supports pluggable look-and-feel, you could have a Windows look in your UNIX environment (if you would ever want that). Check out the numerous subcategories for code examples for each component!



### CHAPTER 3

### Description

Othello is a relatively modern game. Its origins lie in England at the end of the nineteenth century but the game as it is known now was set down in Japan as recently as 1971. It rapidly became popular and the first world championship was organized in 1977. The motto of Othello is **“Easy to learn, difficult to master”** and as such it is ideal for computer implementation. The brief introduction to the game in Appendix A shows how simple the rules are and therefore why the initial coding of a playable game of Othello should, and did, not take long. However, the tactics and strategies of Othello, as explored by Alison, are very complex. Although computer implementations of Othello do exist we strove to implement and combine these existing strategies in a new way.

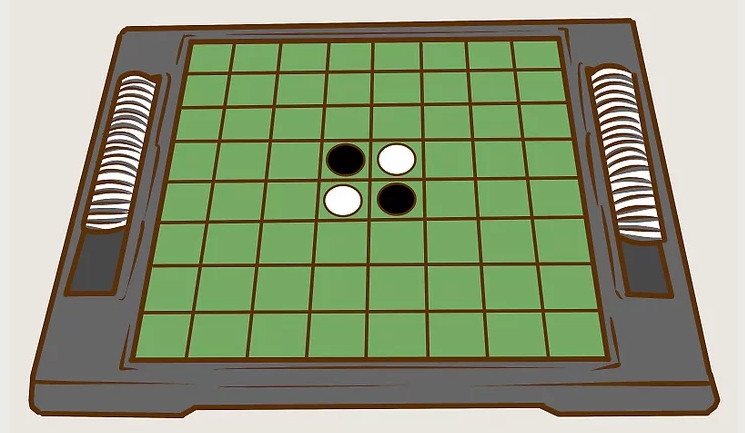
* 1. **Project Aims and Objectives**

The main aims of the project were set out as follows, with the majority of the project time to be spent on the second objective.

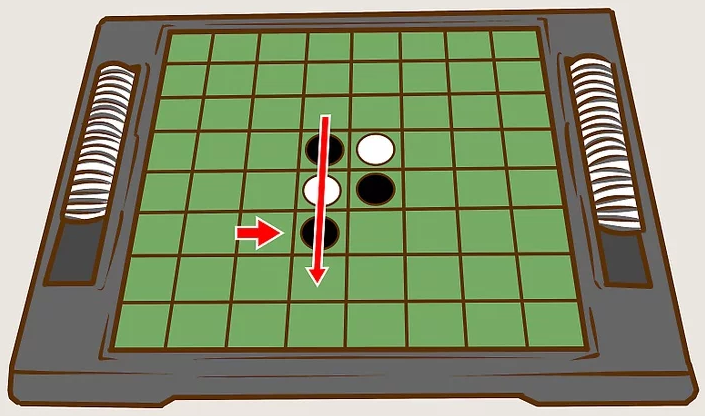
* Produce a computer program implementing the game of Othello.
* Create a Reversi game playing program where the discs on the board can only move accordingly to what is defined as a legal move, by the rules of the game.
* Include a user-friendly graphical interface. It was decided to program the project using Java.

Java classes allow for easy division of the project, providing a clear understanding of the required interfaces. We have tried a lot to achieve all the characteristics of good software. In later version it will be more refined and integrated.

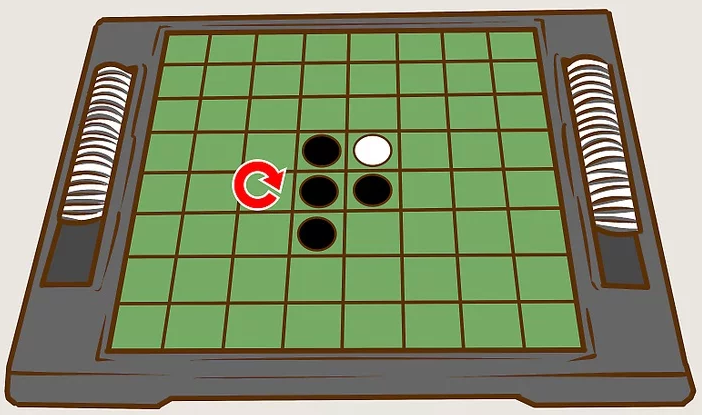
* 1. **Rules of Othello**



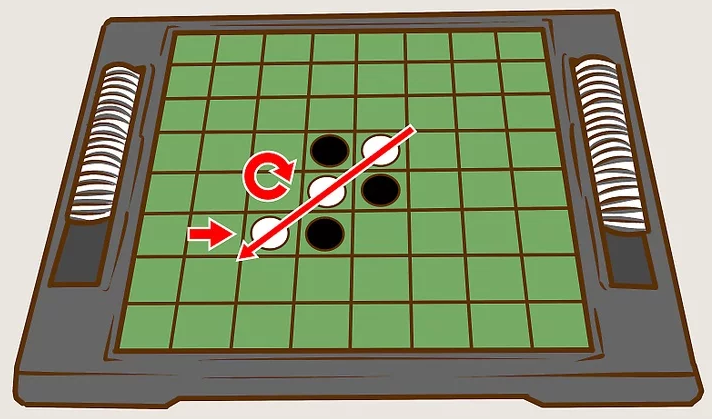
* **Set up the game board and discs.** Othello is played on an 8 x 8 non-checkered board with 64 discs which are black on one side and white on the other. One player plays discs black side up and the other plays white side up. Place 4 discs in the center of the board; 2 with the black side up and 2 with the white side up so that the discs with matching colors touch diagonally. The player with black disks typically goes first.



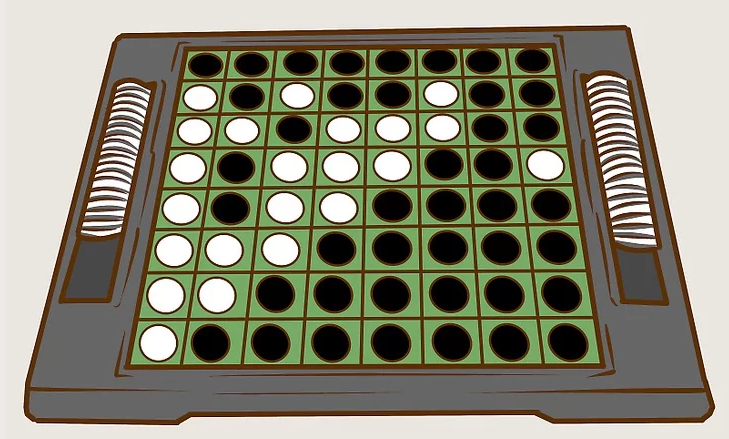
* **Place the first disc in a spot that outflanks an opponent’s disc.** To “outflank” a disc means to surround a row of your opponent’s discs with two of your own discs. A “row” consists of one or more discs that form a line horizontally, vertically or diagonally. For instance, the first player places a black disc in a spot that outflanks a white disc either vertically or horizontally, as it is not possible to outflank the white discs diagonally (assuming black goes first).



* **Flip the outflanked disc to its opposite side.** Once a disc is outflanked, it is flipped to the opposite color and becomes the other player’s disc. Following the example in the previous step, the white disc that is outflanked is turned to black and belongs to the black player.



* **Pass the turn to your opponent.** The opponent now places the second disc in a spot that outflanks at least one of the first player’s discs. Assuming the second player plays the white discs, they would place one of their discs so a row of at least one black disc is framed by two white discs on each side, flipping the outflanked black disks to white. Remember that the row can be horizontal, diagonal or vertical.



* **Continue taking turns placing discs until a legal move isn’t possible.** For a move to be legal, a disc must always be placed in a position where it can outflank a row of the opponent’s discs. If this isn't possible, you must forfeit your turn until you can perform a legal move. If neither player can perform a legal move, usually because all spaces are filled, play ends.
  1. **Implementation**

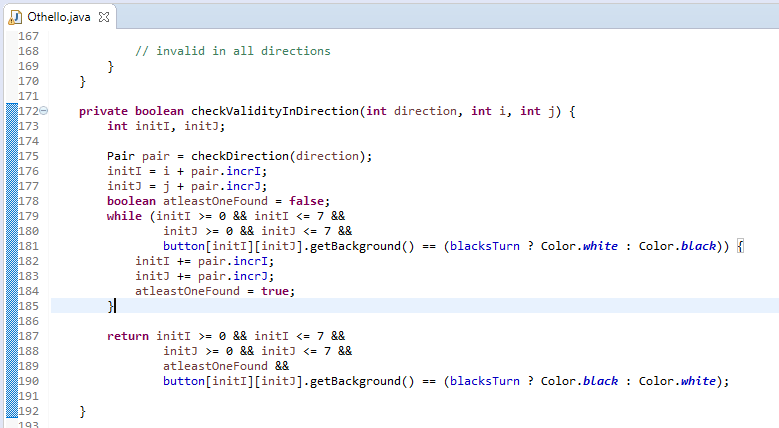
The Othello class’s constructor set ups the game layout, by making use of the JLabels (display area for a short text string or image or both), JTextfield (a lightweight component that allows editing of a single line of text), JButtons (an implementation of a push button). The color and font properties are set in the constructor itself.

The buttons are made responsive using the ActionListener interface. The class that is interested in processing an action event implements this interface and the object created with that class is registered with a component, using the component’s add ActionListener method. When the action event occurs, that object’s action Performed method is invoked.

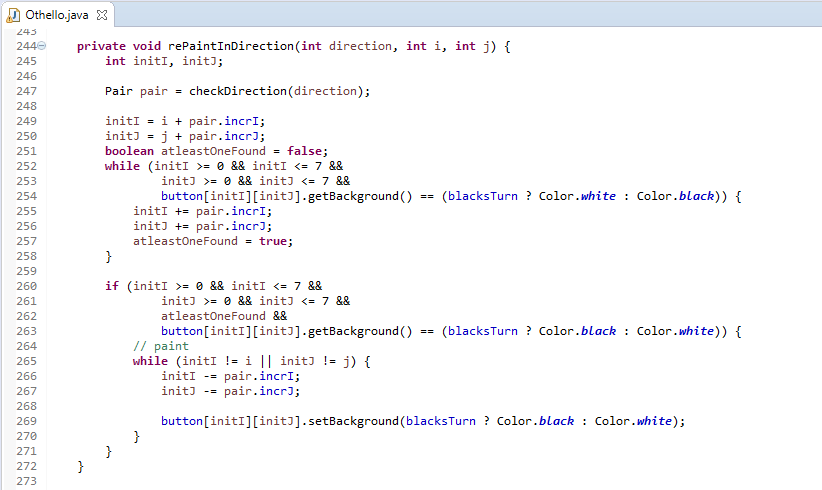
The javax.swing.JFrame class is a type of container which inherits the java.awt.Frame class. JFrame works like the main window where components like labels, buttons, textfields are added to create a GUI.

Unlike Frame, JFrame has the option to hide or close the window with the help of setDefaultCloseOperation(int) method.

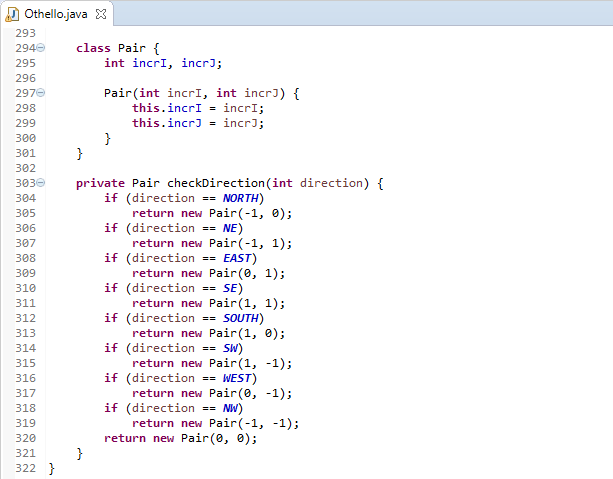
The function checkValidityInDirection() is used to check which cell is valid for making the moves.



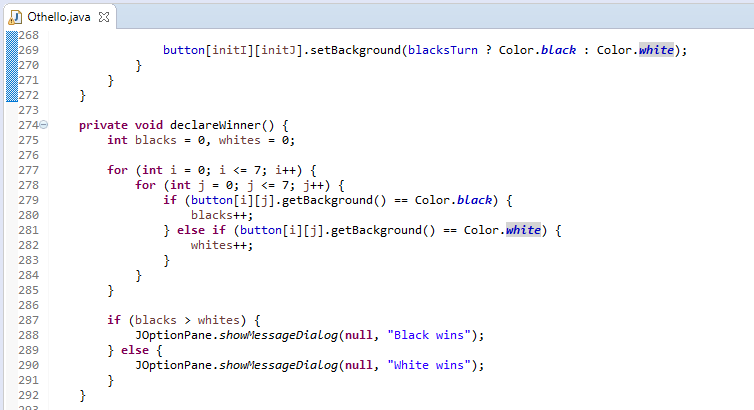
The function rePaintDirection() is used to set the black cells as white and white cells as black according to the move of the player.



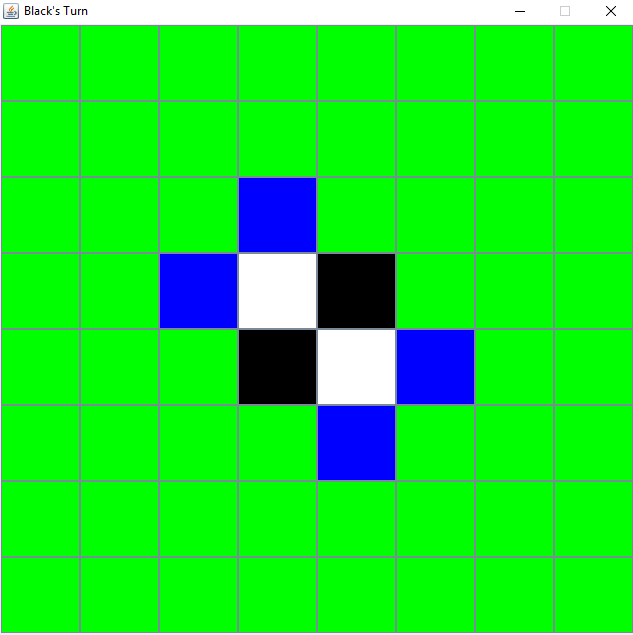
The function checkDirection() is used to check which direction is valid for making the moves.

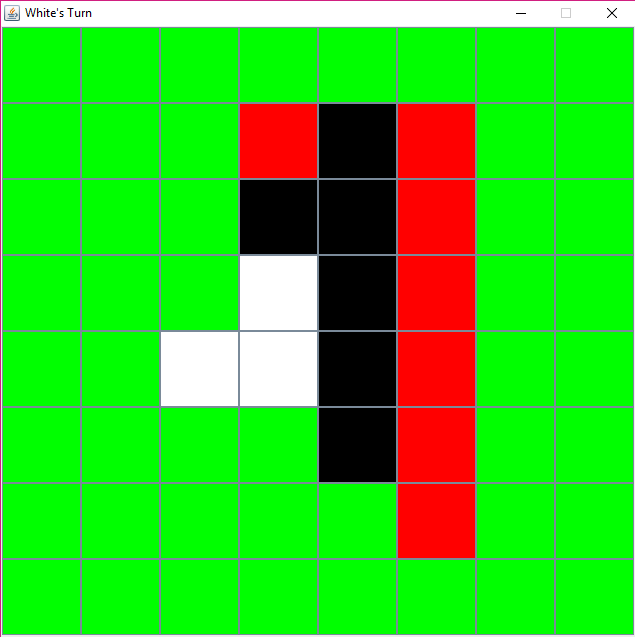


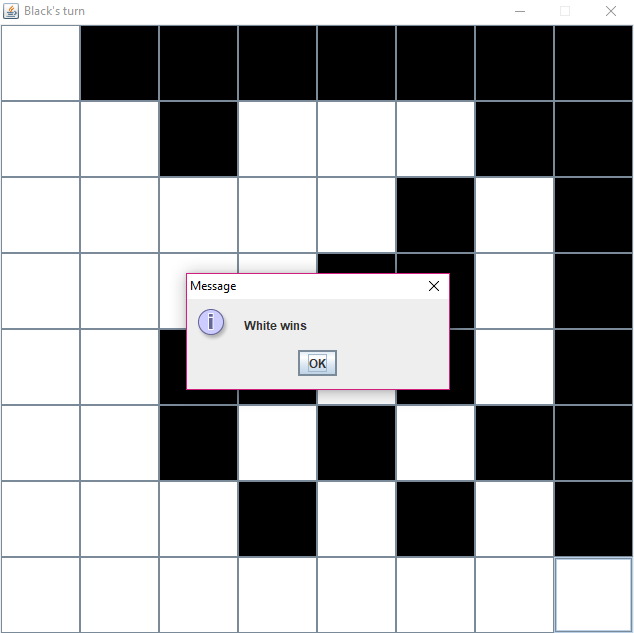
The function declareWinner() is used to declare the winner.

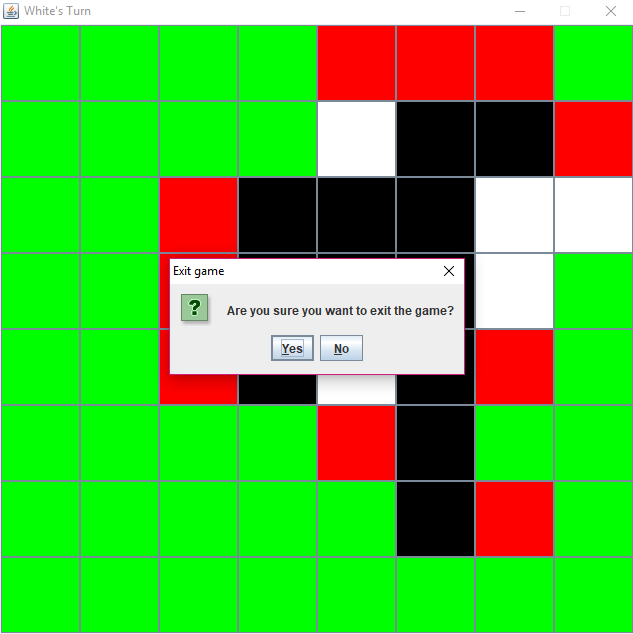


1. **CHAPTER 4**
   1. **OUTPUT SCREENSHOTS**

****

****

****

****

**Conclusion**

**Java** offers the real possibility that most programs can be written in a type-safe language. Designing the above project with object oriented programming concepts and Java programming language is a challenging and interesting at the same time. Efficient exploration and implementing of functionalities of Java has led to practical challenging research activities and development of new standards.

**BIBLIOGRAPHY**

* Complete reference of Java by Herbert Schildt
* Java programming by E Balaguruswamy
* Geeks for Geeks
* Stack Overflow
* Youtube.com
* Wikipedia