

100+ Years of Glorious history inscribed in the
yeomen service to the field of education

Centenary Celebrated Sharnbasveshwar Vidya Vardhak Sangha's

Estd. 2017
ಸ್ಥಾಪನೆ : 2017



Pooja Dr. Sharnbasappa Appaji
Mahadevacharya
Sharnbasveshwar Sanstha
President, Sharnbasveshwar Vidya Vardhak Sangha
Chikballi, Sharnbasva University



Pooja Mahadevi Dr. Sankhagiri E. Appa
Chikballi,
Sharnbasveshwar Vidya Vardhak Sangha
Member of BCC, Sharnbasva University



Pooja Chiranjeevi Doddappa Appa
Mahadevacharya
Sharnbasveshwar Sanstha, Chikballi
Sharnbasveshwar Sanstha, Chikballi

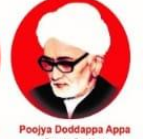
ಶರಣಬಸವ
Sharnbasva



ವಿಶ್ವವಿದ್ಯಾಲಯ
University



Pooja Maheshri Godutai Avaji



Pooja Doddappa Appa
Founder President
Sharnbasveshwar Vidya Vardhak Sangha

Kalaburagi - 585103, Karnataka - India

ಕಲಬುರಗಿ 585 103 ಕರ್ನಾಟಕ - ಭಾರತ

Phone / Fax No. 08472-277852, 277853, 277854, 277855 www.sharnbasvauniversity.edu.in - email : Sharnbasvauniversity@gmail.com

UGC Status: Letter No. F.8-29/2017(CPP-I/PU), Dated 20 Dec. 2017. Enlisted by the University Grants Commission, New Delhi, in the list of Private Universities in India.
A Private University enacted by Govt. of Karnataka as "Sharnbasva University Act. 2012" Karnataka Act No. 17 of 2013. Notification No. ED 144 URC 2016 dated 29/07/2017

Faculty of Engineering & Technology (Co-Ed)

Department of Computer Science &

Engineering

INTERNSHIP REPORT

Girish Patil

USN: SG19CSE053

Semester: 8th

From: 01 APR 2023 To: 25 JUNE 2023

Faculty Advisor

Prof. Anil Dangi

Department of Computer Science & Engineering, Sharnbasva

University,

Kalaburagi, Karnataka.

2023

100+ Years of Glorious history inscribed in the
yeomen service to the field of education

Centenary Celebrated Sharnbasveshwar Vidya Vardhak Sangha's

Estd. 2017
ಸ್ಥಾಪನೆ : 2017



Pooja Dr. Sharnbasappa Appaji
Mahasabha Pradhikarini
Sharnbasveshwar Vardhak Sangha
President, Sharnbasveshwar Vidya Vardhak Sangha
Chancellor, Sharnbasva University



Pooja Koteswari Dr. Gulabayya S. Appa
Chancellor
Sharnbasveshwar Vidya Vardhak Sangha
Member of SSG, Sharnbasva University



Pooja Chiranjeevi Daddappa Appa
Mahasabha 1st Pradhikarini
Sharnbasveshwar Vardhak Sangha
Sharnbasveshwar Vardhak Sangha

ಶರಣಬಸವ
Sharnbasva



ವಿಶ್ವವಿದ್ಯಾಲಯ
University



Pooja Matoshri Godutai Avaji



Pooja Daddappa Appa
Founder President
Sharnbasveshwar Vidya Vardhak Sangha

Kalaburagi - 585103, Karnataka - India

ಕಲಬುರಗಿ 585 103 ಕರ್ನಾಟಕ - ಭಾರತ

Phone / Fax No. 08472-277852, 277853, 277854, 277855 www.sharnbasvauniversity.edu.in - email : Sharnbasvauniversity@gmail.com

UGC Status: Letter No. F.8-29/2017(CPP-I/PU), Dated 20 Dec. 2017. Enlisted by the University Grants Commission, New Delhi, in the list of Private Universities in India.
A Private University enacted by Govt. of Karnataka as "Sharnbasva University Act. 2012" Karnataka Act No. 17 of 2013. Notification No. ED 144 URC 2016 dated 29/07/2017

COMPUTER SCIENCE ENGINEERING DEPARTMENT

CERTIFICATE

Certified that Internship Report entitled "**FINDING VOLUNTARY BLOOD DONOR**"
carried out by **Girish Patil (USN:SG19CSE053)** in partial fulfilment of the requirements
for the degree of B.Tech COMPUTER SCIENCE & ENGINEERING, Sharnbasva
University, Kalaburagi during the academic year 2023. It is certified that all
corrections/suggestions indicated have been incorporated and the report has been approved
as it satisfies the academic requirements in respect prescribed for the said degree.

Marks Details

Sl No	Name of the Student	USN	Maximum Marks	Marks Obtained
1	Girish Patil	SG19CSE053		

Guide

Chairperson

Dean

Names of the examiners

Signature with date

1.

2.

ACKNOWLEDGMENT

Any achievement, be scholastic or otherwise does not depend solely on the individual efforts but also on the guidance, encouragement and co-operation of intellectuals, elders and friends. A number of personalities, in their own capabilities, in their own capacities have helped me in carrying out this project work. I would like to take this opportunity to thank them all.

First of all, I would like to express my immense gratitude to **Dr. Anilkumar Bidve**, Registrar Sharnbasva University, Kalaburagi for his help and inspiration during the tenure of the course.

I also extend my sincere thanks to **Dr. Basavaraj Mathpati**, Registrar (Evaluation) Sharnbasva University, Kalaburagi

I also extend my sincere thanks to **Dr. Laxmi Patil Maka**, Dean Sharnbasva University, Kalaburagi.

I also extend my sincere thanks to **Dr. Shivkumar Jawalagi**, Dean (Faculty of Engineering & Technology - Co.edu) Sharnbasva University, Kalaburagi.

I also extend my sincere thanks to chairman **Dr. Sharanbasappa Madival**, Chairman of the Department, Computer Science Engineering, Faculty of Engineering and Technology- Co.edu, for his constant encouragement.

I also extend my sincere thanks to my Guide **Prof. Anil Dangi**, Department of Computer Science Engineering, Faculty of Engineering and Technology- Co.edu, for this constant encouragement.

I also extend my sense of gratitude and sincere thanks to all the faculty members of Computer Science Engineering and Technology-Co.edu, for their constant encouragement and support.

Girish Patil
(SG19CSE053)

TABLE OF CONTENTS

SL.No	Particular	Pg.No
i.	Brief summary of internship	i
ii.	Completion Certificate	ii

Chapter No.

1.	Internship schedule
2.	Introduction of company and departments
3.	Key Learnings
4.	Documentation
5.	Challenges faced
6.	Conclusion
	References
	Appendix 1: Industry supervisor's report
	Appendix 2: Students log sheet
	Appendix 3: Students feedback form
	Appendix 4: Documentary Evidence's of Internship Such has Screenshots of Online Sessions/Audio-Video Files/Any other details
	Appendix 5: Offer Letter's copies if placed.
	Appendix 6: Internship mail/any other communication letter copy.

BRIEF SUMMARY

Company Name: Ethnus Consultancy Services Private Limited

Company Address: No.151/17/1, SST Chambers, Second Floor, 36th
Cross Road, Jayanagar 5th Block Bengaluru 560041. India

Starting and Completion Dates: 1st April 2023 to 25th June 2023

Department(s) Worked In: Java & Web Essentials

Supervisor's Name: S. RADHA

Position: Manager

Phone Number: Email: reachus@ethnus.com

Authorized Seal and Signatory

CERTIFICATE OF COMPLETION

**PRIVATE &
CONFIDENTIAL**



Explore | Expand | Enrich

ETHNUS CONSULTANCY SERVICES PRIVATE LIMITED

No.151/17/1, SST Chambers, Second Floor, 36th Cross Road, Jayanagar 5th Block,
Bengaluru 560041, India. | reachus@ethnus.com | www.ethnus.com

CIN: U80212KA2010PTC054851

29th June 2023

Dear Girish Patil,

Subject: Internship Completion Letter

This is to certify that Mr. Girish Patil M student from Sharnabasava University had been associated with our organization and completed an internship in Java & Web essentials from 1st April 2023 to 25th June 2023.

We found him to be efficient in satisfying the responsibilities assigned. During this period, we found him to be sincere and efficient with good conduct.

We wish him success in all his future endeavors.

Best Regards

S. Radha
Manager - HRD, Associate Experience
Ethnus Consultancy Services Private Limited



Explore | Expand | Enrich

CHAPTER 1

INTERNSHIP SCHEDULE

Week.No	Tasks
1	Introduction to Training
2	Introduction to Java Programming Language
3	Typecasting, Conditional Statement
4	Break, Continue Statements
5	Arrays, 2D, jagged Arrays
5	Recursion, and examples
6	Oops and their Types
6	Interfaces
7	Java Assessments and Soft Skills
8	Introduction to HTML
9	Cascading Style Sheet
10	JS and SQL

CHAPTER 2

INTRODUCTION OF COMPANY AND DEPARTMENTS

Codemithra is a leading education and placement platform that helps students achieve certification and get placements in large MNCs.

We help students improve their aptitude skills, communication and interpersonal abilities thus making them industry-ready.

This course is designed to teach students and IT professionals the importance of Java in today's world. This course covers Java and Tech essentials for developing projects and content delivery. In this 3months course, you will learn how to use Java Technologies and post completion of session steps, in taking the assessment.

CHAPTER 3

KEY LEARNINGS

SL.No	Departments	Learning
1	DOMAIN	JAVA
2	MANAGER	Completed pending work and interacting with Team Members
3	TEAM MEMBER	Completed the Assigned Work within Time

CHAPTER 4

DOCUMENTATION

What is JDK in detail:

JDK stands for Java Development Kit. It is a software development package provided by Oracle Corporation that includes tools, libraries, and documentation necessary for developing, compiling, and running Java applications. The JDK is an essential component for Java developers as it enables them to write, test, and deploy Java programs.

The JDK consists of several key components, including:

Java Compiler (javac): The Java Compiler is a tool included in the JDK that translates Java source code into bytecode. It takes the source code written by developers, checks it for syntax errors, and produces platform-independent bytecode that can be executed by the Java Virtual Machine (JVM).

Java Virtual Machine (JVM): The JVM is responsible for executing Java bytecode. It is an integral part of the JDK and provides a runtime environment in which Java applications can run. The JVM interprets the bytecode and translates it into machine-specific instructions for the underlying operating system.

Development Tools: The JDK includes a set of development tools that assist in the creation and management of Java applications. These tools include the Java Debugger (jdb) for debugging Java programs, the Java Documentation Generator (javadoc) for generating API documentation, and various other utilities for compiling, packaging, and testing Java code.

Libraries: The JDK provides a comprehensive set of class libraries that offer a wide range of functionality to Java developers. These libraries include standard classes and packages that cover areas such as I/O operations, networking, database connectivity, user interface development, and more. Developers can utilize these libraries to leverage pre-built functionality and accelerate their application development process.

Sample Code and Documentation: The JDK includes extensive documentation and sample code that serves as a valuable resource for developers. The documentation provides detailed information about the Java language, APIs, and tools, while the sample code demonstrates best practices and illustrates how to use various features of the Java platform.

The JDK is typically required for Java development tasks, such as writing, compiling, and testing Java code. It provides a comprehensive set of tools and libraries that empower developers to build robust and scalable Java applications. Additionally, the JDK is platform-specific, meaning that different versions are available for different operating systems (e.g., Windows, macOS, Linux), ensuring compatibility with the target environment.

In summary, the JDK is a software development kit that provides the necessary tools, libraries, and documentation for Java application development. It includes the Java Compiler, JVM, development tools, libraries, sample code, and documentation, enabling developers to create, compile, and run Java programs effectively.

What is JRE in detail :

JRE stands for Java Runtime Environment. It is a software package developed by Oracle Corporation that provides the necessary runtime environment for executing Java applications. JRE is an essential component for running Java programs and is often required to be installed on a system before Java applications can be executed.

The JRE consists of several components that work together to run Java programs efficiently. These components include:

Java Virtual Machine (JVM): The JVM is the core component of the JRE. It is responsible for executing Java bytecode, which is a compiled form of Java source code. The JVM interprets the bytecode and translates it into machine-specific instructions that the underlying operating system can understand and execute.

Class Libraries: The JRE includes a set of pre-compiled class libraries that provide a wide range of functionality to Java applications. These libraries contain reusable code for common tasks such as file I/O, networking, user interface development, and more. Developers can leverage these libraries to build their applications more easily and efficiently.

Java Development Tools: The JRE includes various development tools, such as the Java Compiler (javac) and the Java Debugger (jdb). These tools are used by developers to compile Java source code into bytecode and debug their applications during development.

Java Web Start: Java Web Start is a feature of the JRE that allows users to launch Java applications directly from the web. It provides a seamless way to deploy and update Java applications on client machines without the need for manual installation or updates.

When you install the JRE on your system, it provides the necessary runtime environment to execute Java applications. It allows you to run Java programs without the need for a full Java Development Kit (JDK), which is typically used by developers for writing and compiling Java code. However, it's important to note that the JRE does not include the tools required for Java development.

What is JVM in detail:

JVM stands for Java Virtual Machine. It is a crucial component of the Java platform and plays a central role in executing Java bytecode. The JVM is responsible for interpreting the bytecode and translating it into machine-specific instructions that the underlying operating system can understand and execute. It provides a runtime environment for running Java applications, ensuring platform independence and portability.

Here are some key aspects of the JVM:

Bytecode Execution: Java source code is compiled into platform-independent bytecode. The JVM is responsible for loading and executing this bytecode. It reads the instructions one by one, interprets them, and performs the necessary operations. Some JVM implementations also use Just-In-Time (JIT) compilation techniques to optimize performance by dynamically translating frequently executed bytecode into machine code.

Memory Management: The JVM manages memory allocation and deallocation for Java applications. It includes automatic memory management known as garbage collection, which frees up memory by automatically reclaiming objects that are no longer in use. This relieves developers from manual memory management tasks and helps prevent common memory-related issues like memory leaks and dangling references.

Platform Independence: One of the key advantages of the JVM is its platform independence. Java bytecode is designed to be executed on any system that has a compatible JVM installed. This "write once, run anywhere" principle allows developers to develop Java applications on one platform and deploy them on different platforms without requiring modifications to the code.

Class Loading and Verification: The JVM dynamically loads classes as they are needed during program execution. It locates the required class files, verifies their integrity, and loads them into memory. Class loading also involves resolving dependencies between classes and linking them together to form a complete program.

Security and Sandbox: The JVM provides a built-in security model that helps protect against malicious code execution. It includes a security manager that enforces security policies and restricts access to system resources. The JVM also supports the concept of a sandbox, which isolates untrusted code in a controlled environment, preventing it from accessing sensitive resources or performing harmful actions.

Runtime Monitoring and Management: The JVM provides tools and APIs for monitoring and managing the runtime environment. These tools enable developers to gather information about memory usage, CPU utilization, thread activity, and other runtime statistics. They are useful for performance tuning, debugging, and profiling of Java applications.

It's important to note that the JVM is not limited to Java programming alone. It also supports other languages that target the Java Virtual Machine, such as Kotlin, Scala, and Groovy. This flexibility expands the range of applications that can benefit from the JVM's features and capabilities.

Why is java platform independent:

Java is considered platform independent because of the way it utilizes the Java Virtual Machine (JVM) and the compilation process.

Bytecode and JVM: When Java source code is compiled, it is transformed into platform-independent bytecode rather than machine-specific code. This bytecode is designed to be executed by the JVM, which acts as an intermediary between the bytecode and the underlying operating system. The JVM interprets the bytecode and translates it into machine-specific instructions for the operating system, allowing Java applications to run on different platforms without requiring recompilation.

Write Once, Run Anywhere: Since Java bytecode is platform-independent, Java applications can be developed on one platform (e.g., Windows) and run on any other platform (e.g., macOS, Linux) that has a compatible JVM. This "write once, run anywhere" principle is a key feature of Java, enabling developers to create software that can be

deployed across diverse environments without the need for platform-specific modifications.

Abstraction of Operating System Differences: Java provides a set of standard libraries and APIs (Application Programming Interfaces) that abstract away the differences between operating systems. These libraries offer consistent functionality for tasks such as file I/O, networking, and user interface development. By relying on these standard libraries, developers can write code that remains consistent across multiple platforms, shielding them from low-level platform-specific details.

JVM Adaptation: To ensure platform independence, JVM implementations are developed for different operating systems and architectures. These implementations are responsible for executing Java bytecode on their respective platforms. JVMs are specifically designed to handle the bytecode and provide the necessary runtime environment, including memory management, security, and other runtime features.

It's important to note that while Java itself is platform independent, certain platform-specific features and APIs may not be available or may behave differently across different platforms. In such cases, developers may need to use platform-specific libraries or write platform-specific code to achieve the desired functionality.

How does a java program run:

- The java program is written and saved in a file with the file name as add.java
- This file is also called as source file
- java is the extension
- When Java language is converted using compiler it is converted in Intermediate Level Language or byte code
- Intermediate Level Language means it is nor the high-level language nor the machine level language
- The file name for ILL is add.class
- .class us the extension given to ILL file
- ILL is converted into Machine Level Language using JVM
- JVM will be installed on the OS
- At the time of execution JVM will send one compiler by the name JIT (Just in Time Compiler)

- JIT compiler will convert ILL to Machine Level Language
- MLL is non accessible, hence it has no file name
- MLL will be given to linking loader and all the library files will be linked then the linked file will be loaded to RAM
- RAM will store the loaded file in the form of executable file or image
- While execution only one thing will be checked, that is it will check that whether JVM is installed on OS or not
- Hence Java is platform independent language

Operators:

In Java, operators are symbols that perform specific operations on one or more operands (variables, constants, or expressions) and produce a result. Java provides a wide range of operators that can be classified into several categories:

Arithmetic Operators:

- Addition: "+"
- Subtraction: "-"
- Multiplication: "*"
- Division: "/"
- Modulus (remainder): "%"
- Increment: "++"
- Decrement: "--"

Relational Operators:

- Equal to: "=="
- Not equal to: "!="
- Greater than: ">"
- Less than: "<"
- Greater than or equal to: ">="
- Less than or equal to: "<="

Logical Operators:

- Logical AND: "&&"
- Logical OR: "||"
- Logical NOT: "!"

Assignment Operators:

- Simple assignment: "="
- Addition assignment: "+="
- Subtraction assignment: "-="
- Multiplication assignment: "*="
- Division assignment: "/="
- Modulus assignment: "%="
- And assignment: "&="
- Or assignment: "|="
- Exclusive or assignment: "^="
- Left shift assignment: "<<="
- Right shift assignment: ">>="
- Unsigned right shift assignment: ">>>="

Bitwise Operators:

- Bitwise AND: "&"
- Bitwise OR: "|"
- Exclusive OR: "^"
- Unary bitwise complement: "~"
- Left shift: "<<"
- Right shift: ">>"
- Unsigned right shift: ">>>"

Conditional Operator:

- Conditional (ternary) operator: "condition ? expression1 : expression2"

Datatypes:

In Java, data types define the kind of values that can be stored in variables and the operations that can be performed on those values. Java has two categories of data types:

Primitive Data Types:

- **Boolean:** Represents a boolean value, either true or false.

- **byte:** Represents a signed 8-bit integer value ranging from -128 to 127.
- **short:** Represents a signed 16-bit integer value ranging from -32,768 to 32,767.
- **int:** Represents a signed 32-bit integer value ranging from -2,147,483,648 to 2,147,483,647.
- **long:** Represents a signed 64-bit integer value ranging from -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807.
- **float:** Represents a single-precision 32-bit floating-point value.
- **double:** Represents a double-precision 64-bit floating-point value.
- **char:** Represents a single 16-bit Unicode character.

Reference Data Types:

- **Class:** Represents a class type.
- **Interface:** Represents an interface type.
- **Array:** Represents an ordered collection of elements.
- **Enum:** Represents a fixed set of constants.
- **String:** Represents a sequence of characters.
- **Wrapper Classes:** These are classes that wrap the primitive types and provide additional functionality. Examples include Integer, Double, Boolean, etc.

Main method in java:

***public static void main (String [] args)**

1. Public - it is a access modifier

Main method in java is declared as public because to access from outside the class

2. Static is made because there is no need of creating the object for the main method to access main method
3. Void - it is a datatype or return type which is written just before a function and says no value from the function need to be returned

We can just say void means nothing

* Sting[] args - String is a pre-defined class in java

Pre-defined class in java are written in capital letters

- It us a array type of variable
- It is passed in the form of argument
- To pass any kind of information from main method it is used

Producing the output:

`System.out.print();` - it is an output statement in java through which we can print the variables, expressions and many more contents

Where:

*System - it is a pre-defined class. System is available in java. lang package
java. lang package is a default package in java

*Out - out is a static member of system class

*Print - it is a pre-defined method to print contents in console

Taking input from user and producing the output:

Scanner Class - it is a predefined class in java which is available in java.util package

It is used to take user input

Rule 1 - if we use Scanner class, we must have to create object for Scanner class

Syntax: `Scanner var_name = new Scanner (System.in);`

where system is pre-defined class and in is used because we take input, in is a member of pre-defined class

Rule 2 - Scanner class methods:

1 `nextLine();` : to take string input

2 `nextInt();` : to take integer input

3 `nextFloat();` : to take floating point value input

4 `nextBoolean();` : to take boolean value input

5 `nextDouble();` : to take double value input

Rule 3 - import Scanner class package at the top line of package

Syntax - `import java.util.Scanner;`

Class : it is a collection of objects and it doesn't take any space on memory.

* It is also called as blue print or a logical entity.

* The class can be of two types :

1. **Pre-defined class** - a class which is already defined

For ex - Scanner , Console , System , String

2. **User-defined class** - a class which is defined by the user

For ex - Dog , Demo , Test

Simply a class is a collection of Data Members and Methods

Object : it is a instance of a class that executes the class

Once the object is created it takes up the space like other variables in the memory

Syntax: class_name object_name = new class_name();

where class_name is the name of the class in which object is created

object_name is the reference variable of the object

'=' is the assignment operator

class_name() is the constructor....

Variables - name of the memory location

It is user defined name which is given by the user

It can store any type of values

Types of variables in java

1 local variable

2 instance variable

3 static variable

4 reference variable

1 **Local variable** - a variable which is declared inside the body of the method or method parameter

Local variable is accessed directly

Accessable only within the variable

2 **Instance variable** - A variable which is declared inside a class but outside the method

Instance variable can be accessed through creating the objects

Instance variable are accessable throughout the class

3 **Static variable** - A variable which is created with the help of using a static keyword

Syntax : static int a;

where the value of 'a' will be fixed

Static variables are declared anywhere in the program

Static variables are accessed through the class

4 Reference variable - A variable which is used to store the reference address of the object

Packages - A package arrange number of classes , interfaces and subclasses of the same type into particular groups

Note - package is nothing but like a folder in windows

There are two types of packages:

- 1 pre-defined
- 2 User-defined

Ex for pre-defined classes are :

- *java.lang
- *java.util
- *java.io
- *java.aut
- *java.sql etc

Ex for user-defined classes are :

- *package add
- *package myPack

User-defined names will be given by user where the names can be given anything depending on user

Access modifiers :

There are basically 4 access modifiers

- 1 Private
- 2 Default
- 3 Protected
- 4 Public

- 1 Private - when we declare as private , it is accessible within the same class
- 2 Default - when we declare as Default , it is accessible with the same class , same package
- 3 Protected - when we declare as Protected , it is accessible with the same class , outside the package by subclass
- 4 Public - when we declare as Public , it is accessible with the same class , same package , outside the package by subclass , outside the package

Advantage of Access Modifiers is : Reuseability , Security , Fast Searching , Hiding
Disadvantage is - parameters cannot be passed

Method - it is a block or group of code , which take input from user process it and give output

Method runs only when it is called

Method is made only inside the class

Types of methods are -

1 Pre-defined

- * print();
- * sort();
- * nextInt();

2 User-defined

- * add();
- * sub();
- * show();

```
return_type function_name()
{
//Statements
}
```

Method is used to reduce line of code

Readability is effective

Repetition can be done of method how many times it is called

Flow control - it describes the order in which statements will be executed at runtime

Control statements - decision to be made by evaluating a given expression as true or false, such expression involves relational and logical operators depending upon the outcome of the decision, program execution proceeds in the direction of one or another

Types of control statements -

if statement: Executes a block of code if a given condition is true.

if-else statement: Executes one block of code if a condition is true and another block if it is false.

if-else-if ladder: Executes different blocks of code based on multiple conditions.

Looping Statements:

for loop: Repeats a block of code for a specified number of times or for each element in a collection.

while loop: Repeats a block of code as long as a condition is true.

do-while loop: Repeats a block of code at least once, then continues as long as a condition is true.

Control Transfer Statements:

Break statement: Terminates the execution of a loop or switch statement.

continue statement: Skips the remaining code in a loop and proceeds to the next iteration.

return statement: Terminates the execution of a method and returns a value to the caller.

Switch statement: Allows multiway branching based on the value of an expression.

These control statements provide essential mechanisms for decision-making, iteration, and control flow within programs, enabling developers to create more complex and dynamic applications.

Arrays:

Arrays in Java are used to store multiple values of the same data type in a contiguous memory location. They provide a convenient way to work with collections of elements, such as a list of numbers, names, or objects. Here are some important aspects of arrays in Java:

Declaration and Initialization: An array is declared by specifying the data type of its elements, followed by square brackets [], and then the array name. Array initialization involves specifying the size of the array and assigning values to its elements.

Example of declaring and initializing an array of integers:

```
int[] numbers; // declaration
numbers = new int[5]; // initialization with size 5
```

```
int[] numbers = { 1, 2, 3, 4, 5 }; // declaration and initialization in one line
```

Accessing Array Elements: Array elements are accessed using their index, which starts at 0 and goes up to length - 1. You can read or modify individual elements by specifying the index within square brackets.

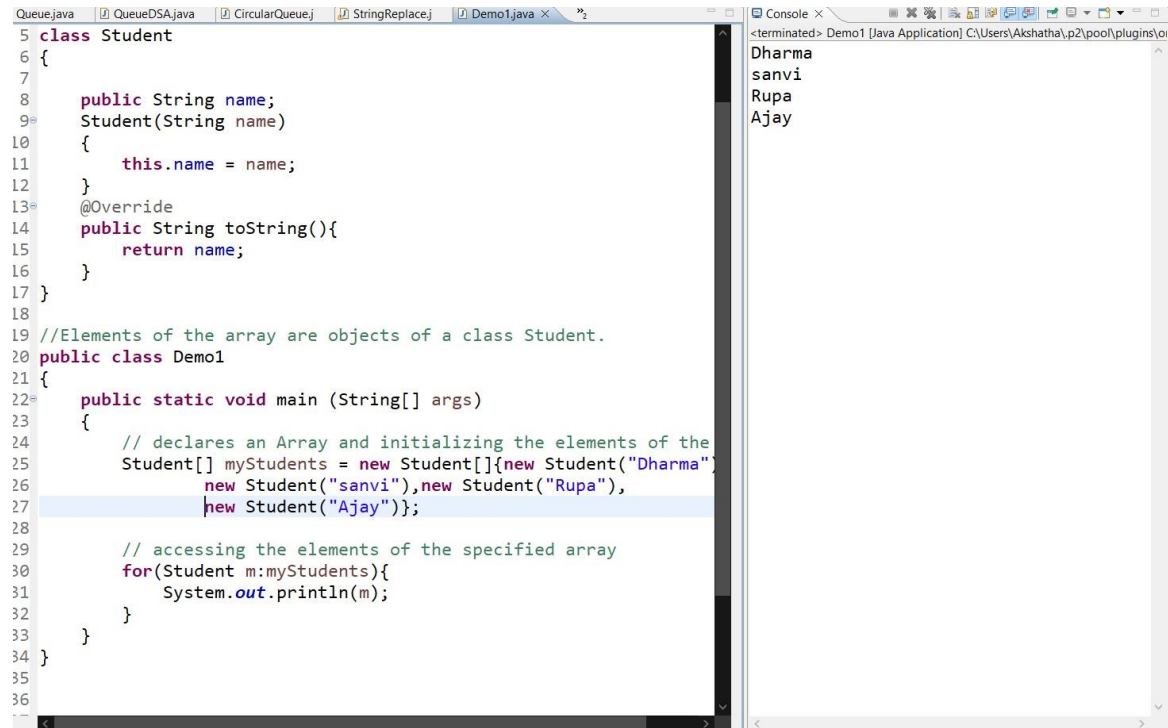
Example:

```
int[] numbers = { 1, 2, 3, 4, 5 };
int firstNumber = numbers[0]; // Accessing the first element
numbers[2] = 10; // Modifying the third element
```

Array Length: The length of an array can be obtained using the length property. It represents the number of elements in the array.

Example:

```
int[] numbers = { 1, 2, 3, 4, 5 };
int length = numbers.length; // length is 5
```



```
5 class Student
6 {
7
8     public String name;
9     Student(String name)
10    {
11        this.name = name;
12    }
13    @Override
14    public String toString(){
15        return name;
16    }
17 }
18
19 //Elements of the array are objects of a class Student.
20 public class Demo1
21 {
22     public static void main (String[] args)
23     {
24         // declares an Array and initializing the elements of the
25         Student[] myStudents = new Student[]{new Student("Dharma"),
26         new Student("sanvi"),new Student("Rupa"),
27         new Student("Ajay")};
28
29         // accessing the elements of the specified array
30         for(Student m:myStudents){
31             System.out.println(m);
32         }
33     }
34 }
35
36
```

Console X
<terminated> Demo1 [Java Application] C:\Users\Akshatha\p2\pool\plugins\o
Dharma
sanvi
Rupa
Ajay

Iterating over an Array: You can iterate over the elements of an array using loops, such as for or foreach.

Example:

```
int[] numbers = {1, 2, 3, 4, 5};
for (int i = 0; i < numbers.length; i++) {
    System.out.println(numbers[i]);
}
```

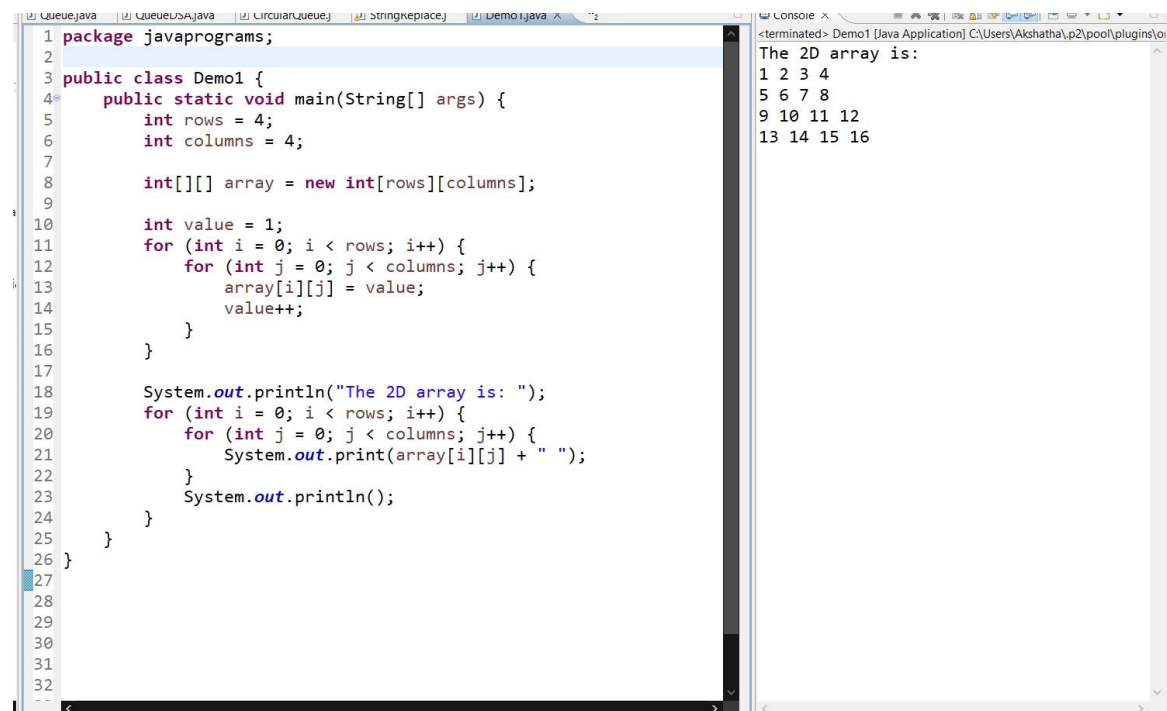
```
// Enhanced for loop (foreach)
for (int number : numbers) {
    System.out.println(number);
}
```

Multidimensional Arrays: Java supports multidimensional arrays, allowing you to create arrays of arrays. Common examples include 2D arrays (arrays of arrays) and jagged arrays (arrays with different lengths).

Example of a 2D array:

```
int[][] matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
```


`int element = matrix[1][2]; // Accessing the element at row 1, column 2 (value 6)`



The screenshot shows a Java IDE with two windows. The left window displays the source code for a class named `Demo1`. The code creates a 4x4 2D array, fills it with values from 1 to 16 in row-major order, and prints the array. The right window shows the console output, which matches the code's execution: it prints "The 2D array is:" followed by a 4x4 grid of numbers from 1 to 16.

```
1 package javaprograms;
2
3 public class Demo1 {
4     public static void main(String[] args) {
5         int rows = 4;
6         int columns = 4;
7
8         int[][] array = new int[rows][columns];
9
10        int value = 1;
11        for (int i = 0; i < rows; i++) {
12            for (int j = 0; j < columns; j++) {
13                array[i][j] = value;
14                value++;
15            }
16        }
17
18        System.out.println("The 2D array is: ");
19        for (int i = 0; i < rows; i++) {
20            for (int j = 0; j < columns; j++) {
21                System.out.print(array[i][j] + " ");
22            }
23            System.out.println();
24        }
25    }
26 }
27
28
29
30
31
32
--
```

<terminated> Demo1 [Java Application] C:\Users\Akshatha\p2\pool\plugins\o
The 2D array is:
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16

Array Manipulation: Java provides utility methods in the Arrays class for various array operations, such as sorting, searching, copying, and filling arrays. These methods can be useful when working with arrays.

Example:

```
int[] numbers = {5, 2, 8, 1, 9};
```

```
Arrays.sort(numbers); // Sorts the array in ascending order
```

```
int index = Arrays.binarySearch(numbers, 8); // Searches for the index of value 8
```

Arrays in Java have a fixed size once they are created, so you cannot dynamically resize them. If you need a resizable collection, you can consider using ArrayList or other dynamic data structures provided by the Java Collections Framework.

String:

In Java, a string is a sequence of characters that represents text. Strings are widely used in Java programs for storing and manipulating textual data. Java provides a built-in class called String to work with strings. Here are some important aspects of strings in Java:

.

String Declaration and Initialization: You can declare and initialize a string using the String class. There are several ways to create and initialize strings:

```
String str1 = "Hello"; // Using string literals
String str2 = new String("World"); // Using the `new` keyword
String str3 = ""; // Empty string
String str4 = null; // Null string
```

String Concatenation: Java provides the + operator to concatenate strings together. You can concatenate two or more strings to create a new string.

```
String greeting = "Hello" + "World"; // Concatenating two strings
String name = "John";
String message = "Welcome, " + name + "!"; // Concatenating a string and a variable
```

String Length: The length of a string can be obtained using the length() method. It returns the number of characters in the string.

```
String str = "Hello";
int length = str.length(); // length is 5
```

String Comparison: You can compare strings for equality using the equals() method, which compares the content of the strings. The == operator can also be used for string comparison, but it checks for reference equality (i.e., whether the two string references point to the same object in memory).

```
String str1 = "Hello";
String str2 = "Hello";
boolean isEqual = str1.equals(str2); // true
```

```
String str3 = new String("Hello");
boolean isSameReference = (str1 == str3); // false
```

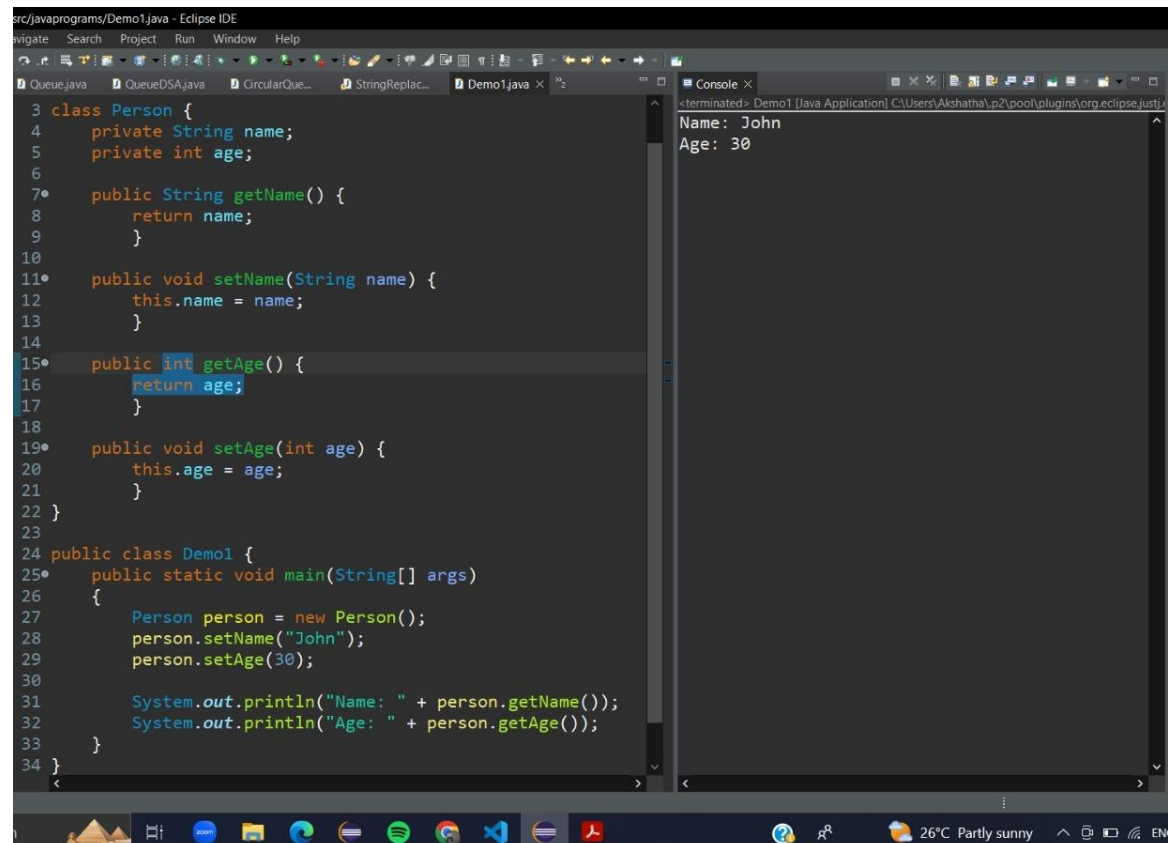
Immutability: In Java, strings are immutable, meaning their values cannot be changed once they are created. When you perform operations on strings, such as concatenation or manipulation, a new string object is created. This immutability property ensures string integrity and simplifies string handling in concurrent environments.

Encapsulation: the method of wrapping up of data and functions that is variables and methods into a single unit

It is also known as information hiding concept as the data will not be accessible to the outside of the class

Only those functions which are wrapped in the class can access

All the four access specifiers are used for the security and hiding of data



The screenshot shows the Eclipse IDE with a Java project. The main editor displays the following code:

```
3 class Person {
4     private String name;
5     private int age;
6
7     public String getName() {
8         return name;
9     }
10
11    public void setName(String name) {
12        this.name = name;
13    }
14
15    public int getAge() {
16        return age;
17    }
18
19    public void setAge(int age) {
20        this.age = age;
21    }
22 }
23
24 public class Demo1 {
25    public static void main(String[] args)
26    {
27        Person person = new Person();
28        person.setName("John");
29        person.setAge(30);
30
31        System.out.println("Name: " + person.getName());
32        System.out.println("Age: " + person.getAge());
33    }
34 }
```

The console on the right shows the output of the program:

```
<terminated> Demo1 [Java Application] C:\Users\Akshatha\p2\pool\plugins\org.eclipse.justi...
Name: John
Age: 30
```

Inheritance: when we construct a new class from existing class in such a way that the new class access all the features and properties of existing class is called inheritance

Key properties:

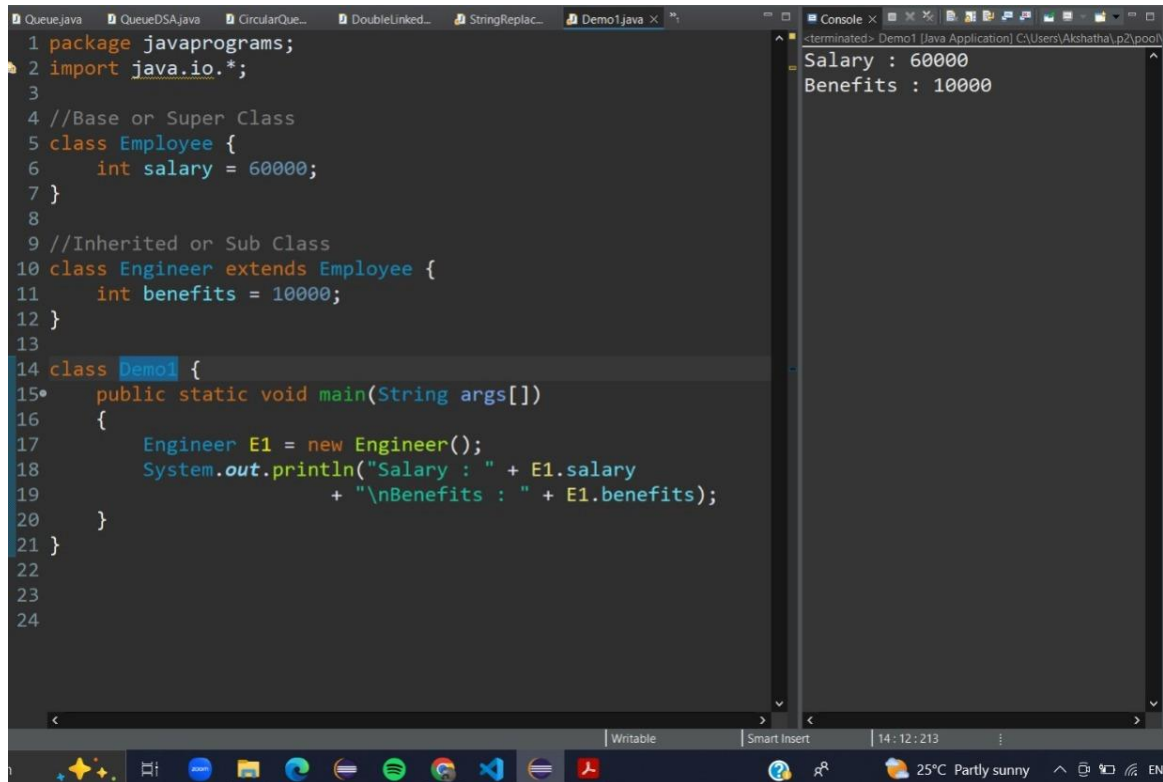
- 1 Extends keyword is used to perform inheritance
- 2 It provides code reuse-ability
- 3 We can't access private members of class through inheritance

Subclass contains all the features of super class

Method overriding only possible through inheritance

Types of Inheritance:

Single level Inheritance: in this type of inheritance there will be only 2 classes one super class and one subclass

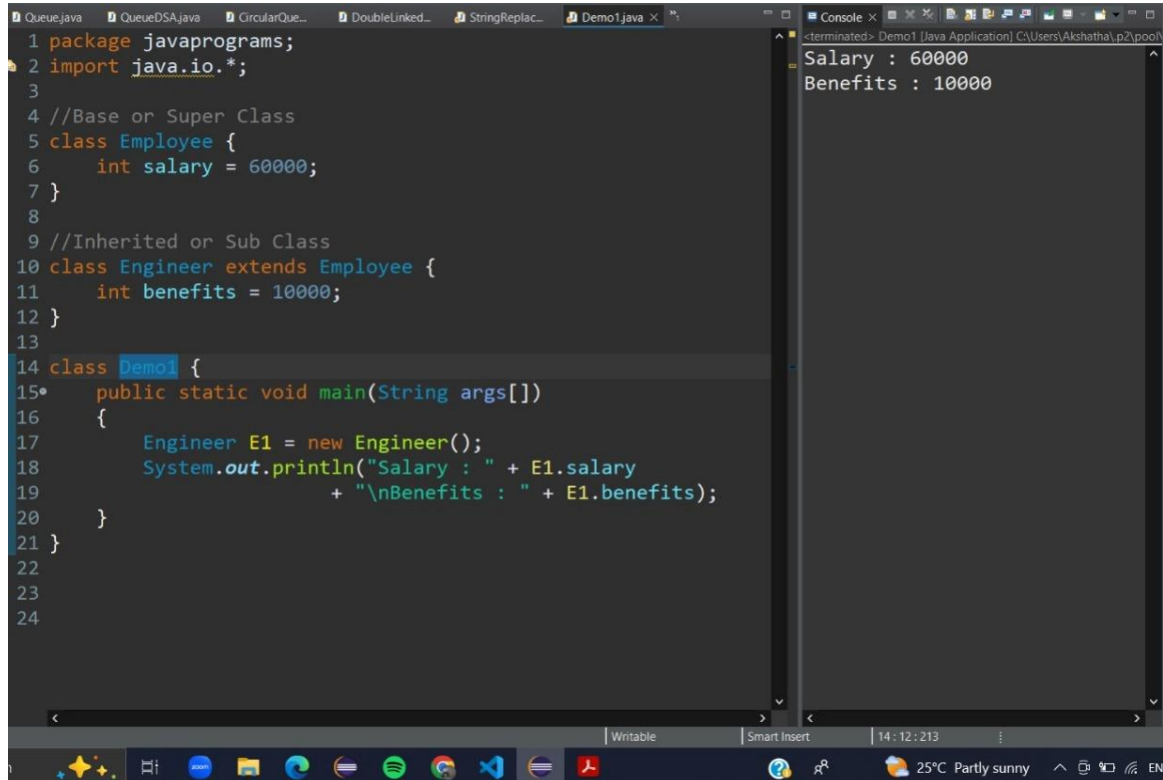


```
1 package javaprograms;
2 import java.io.*;
3
4 //Base or Super Class
5 class Employee {
6     int salary = 60000;
7 }
8
9 //Inherited or Sub Class
10 class Engineer extends Employee {
11     int benefits = 10000;
12 }
13
14 class Demo1 {
15     public static void main(String args[])
16     {
17         Engineer E1 = new Engineer();
18         System.out.println("Salary : " + E1.salary
19                             + "\nBenefits : " + E1.benefits);
20     }
21 }
22
23
24
```

Console Output:

```
<terminated> Demo1 [Java Application] C:\Users\Akshatha\p2\pool\
Salary : 60000
Benefits : 10000
```

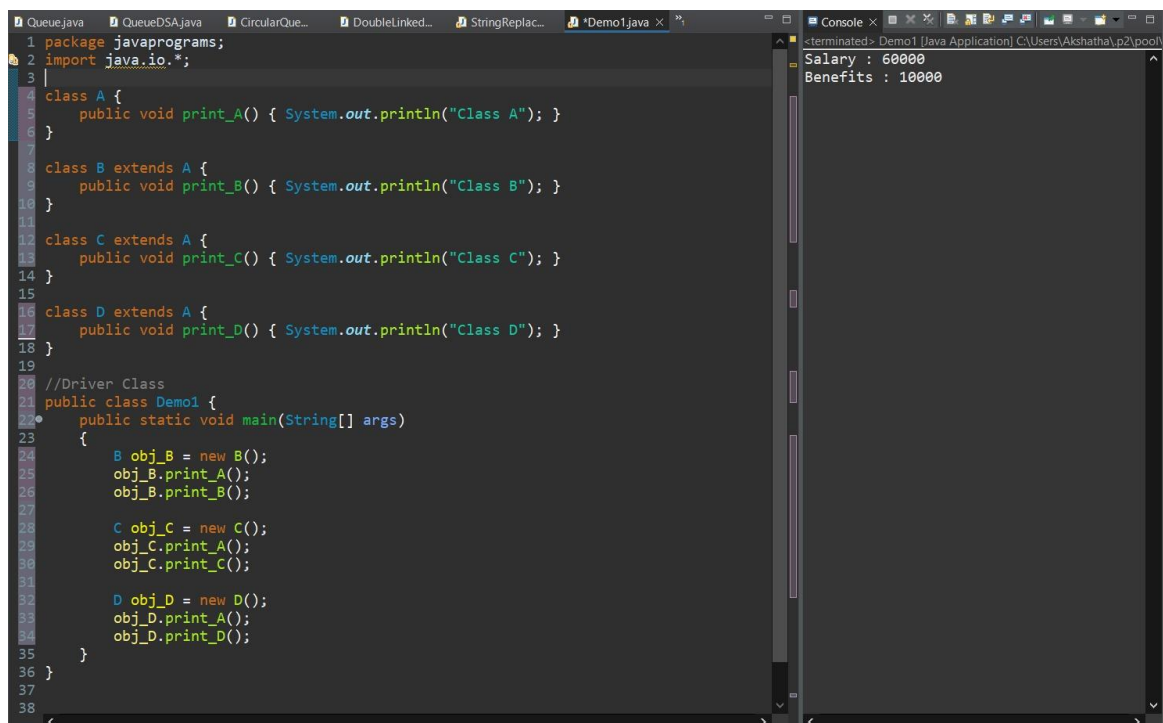
Multilevel Inheritance: in this type of inheritance there will be multiple parents at different level



```
1 package javaprograms;
2 import java.io.*;
3
4 //Base or Super Class
5 class Employee {
6     int salary = 60000;
7 }
8
9 //Inherited or Sub Class
10 class Engineer extends Employee {
11     int benefits = 10000;
12 }
13
14 class Demo1 {
15     public static void main(String args[])
16     {
17         Engineer E1 = new Engineer();
18         System.out.println("Salary : " + E1.salary
19                             + "\nBenefits : " + E1.benefits);
20     }
21 }
22
23
24
```

Salary : 60000
Benefits : 10000

Hierarchical Inheritance: in this type of inheritance there will be only one parent many child



```
1 package javaprograms;
2 import java.io.*;
3
4 class A {
5     public void print_A() { System.out.println("Class A"); }
6 }
7
8 class B extends A {
9     public void print_B() { System.out.println("Class B"); }
10 }
11
12 class C extends A {
13     public void print_C() { System.out.println("Class C"); }
14 }
15
16 class D extends A {
17     public void print_D() { System.out.println("Class D"); }
18 }
19
20 //Driver Class
21 public class Demo1 {
22     public static void main(String[] args)
23     {
24         B obj_B = new B();
25         obj_B.print_A();
26         obj_B.print_B();
27
28         C obj_C = new C();
29         obj_C.print_A();
30         obj_C.print_C();
31
32         D obj_D = new D();
33         obj_D.print_A();
34         obj_D.print_D();
35     }
36 }
37
38
```

Class A
Class B
Class C
Class D

4 Multiple Inheritance: in this type of inheritance there will be multiple parents at same level

It is not supported in java due to ambiguity but supported by interfaces but not by class

5 Hybrid Inheritance: in this type of inheritance there will be the combination of Hierarchical and Multiple

Hybrid Inheritance is also not possible in java because it supports Multiple Inheritance

Polymorphism: in this the same object having different behaviour or also same method name doing different operations will be performed

There are two types of Polymorphism :

1 Compile Time Polymorphism or Static Polymorphism

2 RunTime Polymorphism or Dynamic Polymorphism

1 Static Polymorphism : a polymorphism which exists at the time of compilation

Compile time polymorphism is achieved through method overloading

Method overloading: when the class contain more than one method with same name and different parameters

Syntax : `return_type method_name(parameter1);`

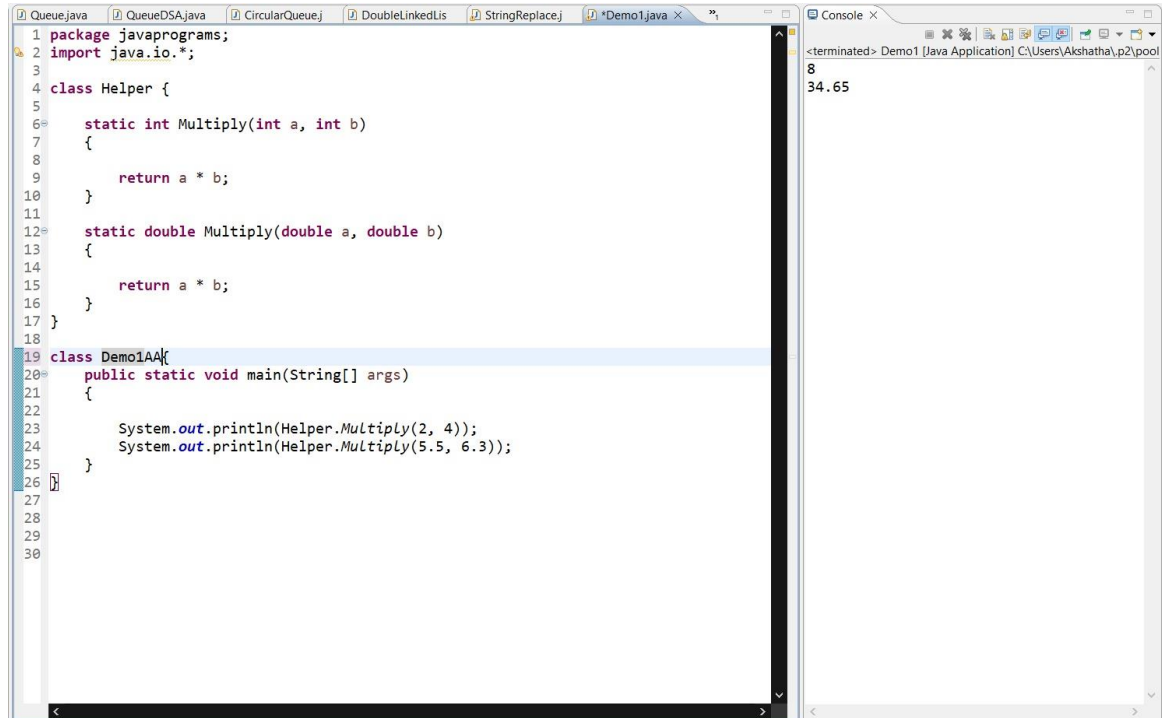
`return_type method_name(parameter1 , parameter2);`

Run Time Polymorphism: a polymorphism which exists at the time of execution of program

run time polymorphism is achieved through method overriding

Method overriding: when a class contain more than one method with same name and same parameters

Method overriding cannot be performed without inheritance so extend keyword is used



The screenshot shows an IDE with a Java file named `*Demo1.java`. The code defines a `Helper` class with two static methods, `Multiply`, one for `int` and one for `double`. A `Demo1AA` class extends `Helper` and calls both methods in its `main` method. The console output shows the results of these calls: `8` and `34.65`.

```
1 package javaprograms;
2 import java.io.*;
3
4 class Helper {
5
6     static int Multiply(int a, int b)
7     {
8
9         return a * b;
10    }
11
12    static double Multiply(double a, double b)
13    {
14
15        return a * b;
16    }
17 }
18
19 class Demo1AA{
20     public static void main(String[] args)
21     {
22
23         System.out.println(Helper.Multiply(2, 4));
24         System.out.println(Helper.Multiply(5.5, 6.3));
25     }
26 }
27
28
29
30
```

Console Output:

```
<terminated> Demo1 [Java Application] C:\Users\Akshatha\p2\pool
8
34.65
```

Abstraction: it is a process of hiding the implementation details from the user

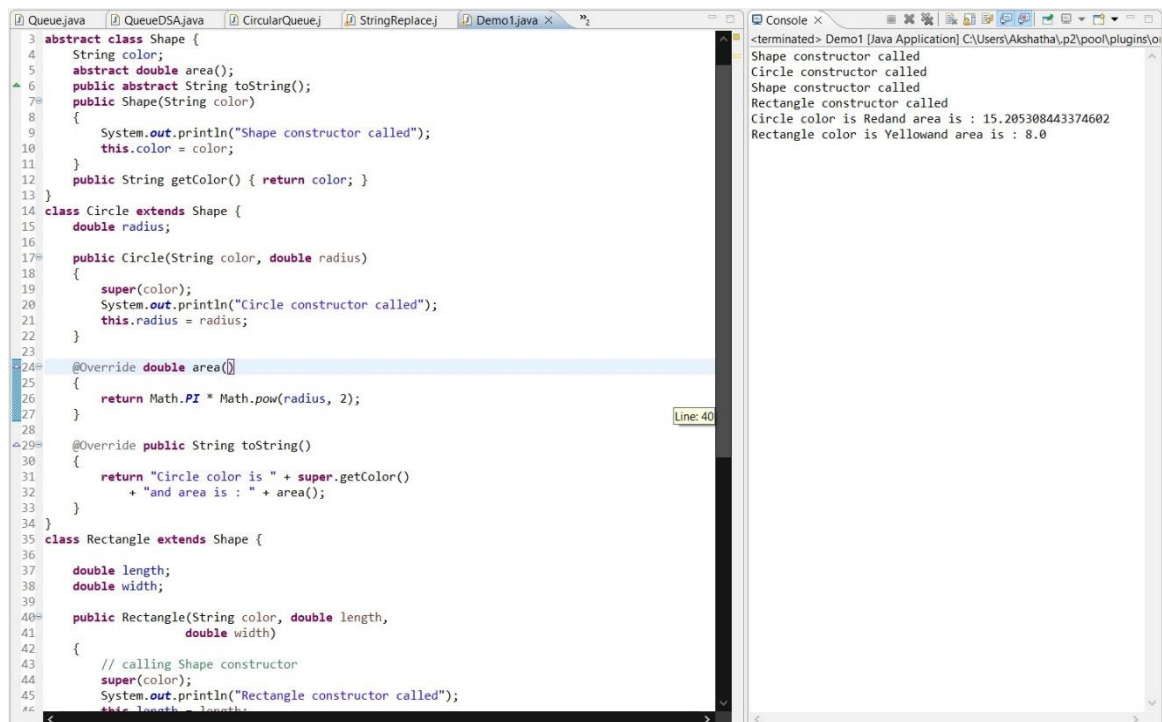
To create abstract method there should be no method definition and abstract keyword should be placed in method signature

If a class have at least one abstract method then the whole class is known as abstract class

Abstract class can contain both abstract class and non-abstract class

An object cannot be created for an abstract class

It can have abstract method or non-abstract method



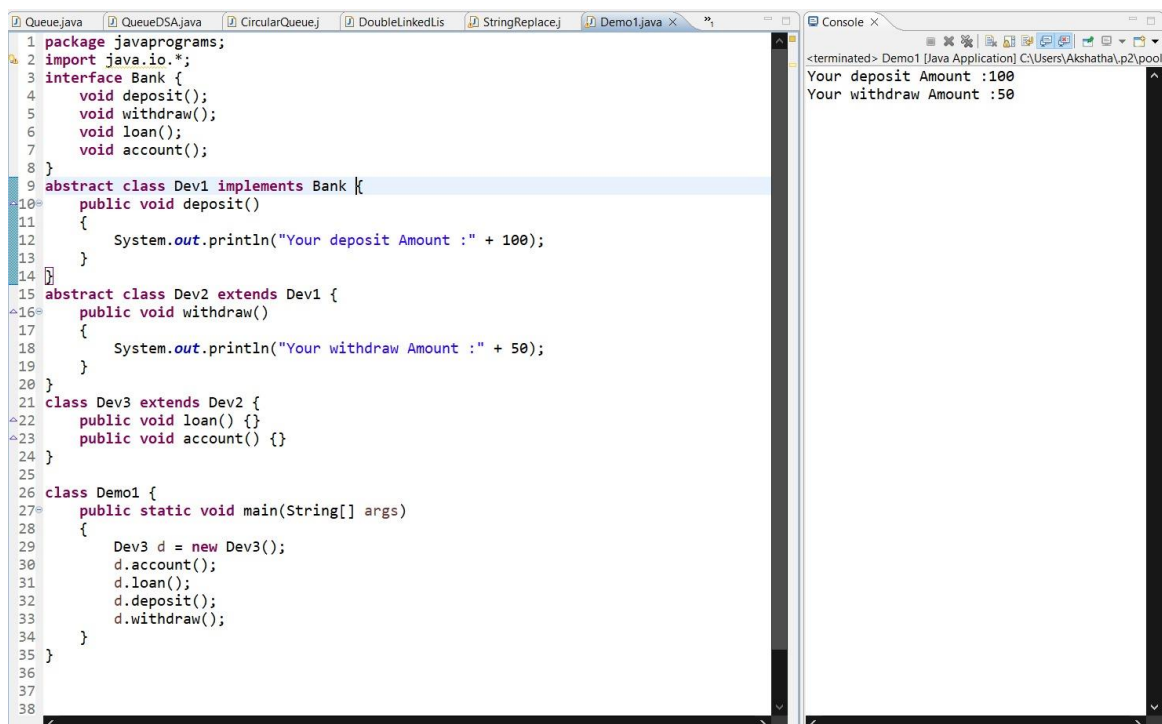
```
3 abstract class Shape {
4     String color;
5     abstract double area();
6     public abstract String toString();
7     public Shape(String color)
8     {
9         System.out.println("Shape constructor called");
10        this.color = color;
11    }
12    public String getColor() { return color; }
13 }
14 class Circle extends Shape {
15     double radius;
16
17     public Circle(String color, double radius)
18     {
19         super(color);
20         System.out.println("Circle constructor called");
21         this.radius = radius;
22     }
23
24     @Override double area()
25     {
26         return Math.PI * Math.pow(radius, 2);
27     }
28
29     @Override public String toString()
30     {
31         return "Circle color is " + super.getColor()
32             + "and area is : " + area();
33     }
34 }
35 class Rectangle extends Shape {
36     double length;
37     double width;
38
39     public Rectangle(String color, double length,
40                     double width)
41     {
42         // calling Shape constructor
43         super(color);
44         System.out.println("Rectangle constructor called");
45         this.length = length;
46         this.width = width;
47     }
48 }
```

```
<terminated> Demo1 [Java Application] C:\Users\Akshatha\p2\pool\plugins\o
Shape constructor called
Circle constructor called
Shape constructor called
Rectangle constructor called
Circle color is Redand area is : 15.205308443374602
Rectangle color is Yellowand area is : 8.0
```

Interfaces: interfaces are also similar to the class

All the data members in interfaces will be public and abstract in nature

To relate between interfaces and class implements keyword is used



```
1 package javaprograms;
2 import java.io.*;
3 interface Bank {
4     void deposit();
5     void withdraw();
6     void loan();
7     void account();
8 }
9 abstract class Dev1 implements Bank {
10     public void deposit()
11     {
12         System.out.println("Your deposit Amount : " + 100);
13     }
14 }
15 abstract class Dev2 extends Dev1 {
16     public void withdraw()
17     {
18         System.out.println("Your withdraw Amount : " + 50);
19     }
20 }
21 class Dev3 extends Dev2 {
22     public void loan() {}
23     public void account() {}
24 }
25
26 class Demo1 {
27     public static void main(String[] args)
28     {
29         Dev3 d = new Dev3();
30         d.account();
31         d.loan();
32         d.deposit();
33         d.withdraw();
34     }
35 }
36
37
38
```

```
<terminated> Demo1 [Java Application] C:\Users\Akshatha\p2\pool
Your deposit Amount :100
Your withdraw Amount :50
```


Exception Handling:

Exception handling in Java allows you to handle and manage runtime errors and exceptional conditions that may occur during the execution of a program. By using exception handling, you can gracefully recover from errors and take appropriate actions, such as displaying an error message, logging the error, or terminating the program in a controlled manner.

Here's an overview of how exception handling works in Java:

Exception Types: Java provides a hierarchy of exception types. The root of this hierarchy is the class `Throwable`, which has two main subclasses:

- **Exception:** Represents exceptional conditions that can be caught and handled by the program.
- **Error:** Represents severe and unrecoverable errors that usually indicate serious problems in the JVM or the underlying system.

Handling Exceptions: To handle exceptions, you use the try-catch statement. The try block contains the code that may throw an exception, and the catch block catches and handles the exception. You can have multiple catch blocks to handle different types of exceptions.

The general syntax is:

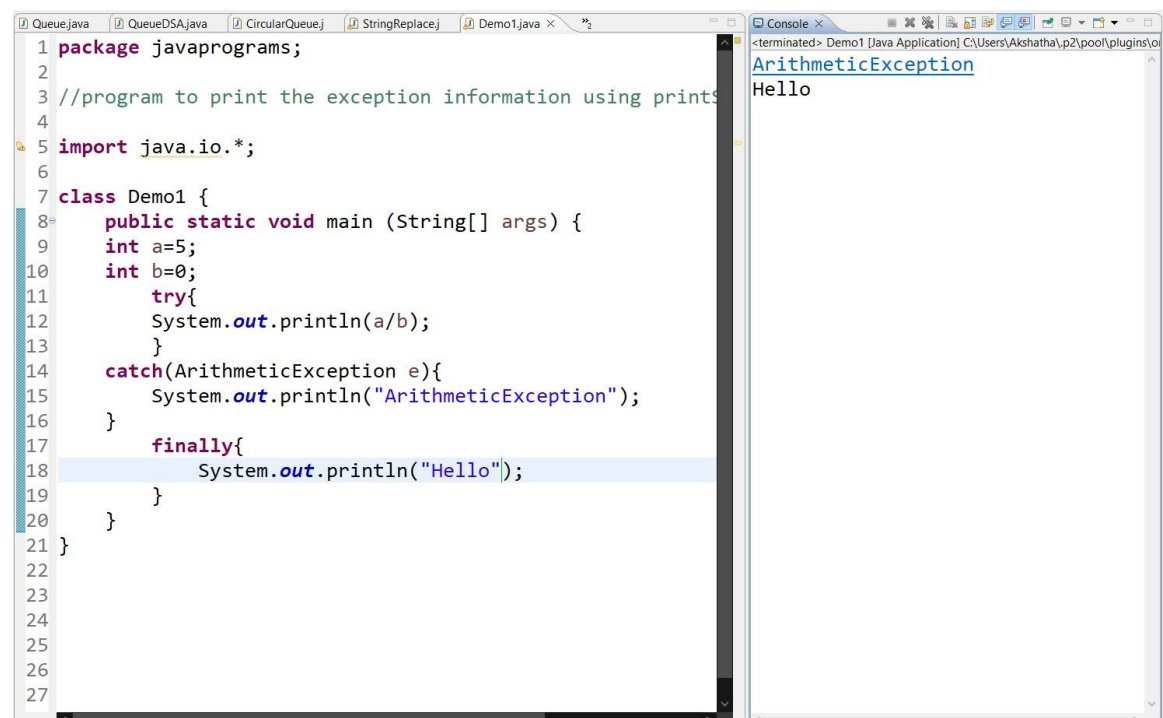
```
try {  
    // Code that may throw an exception  
} catch (ExceptionType1 e1) {  
    // Exception handling code for ExceptionType1  
} catch (ExceptionType2 e2) {  
    // Exception handling code for ExceptionType2  
} finally {  
    // Optional block that is executed regardless of whether an exception is thrown or not  
}
```

Throwing Exceptions: You can explicitly throw exceptions using the throw keyword. This allows you to create and throw your own exceptions or propagate exceptions to the calling code.

```
throw new ExceptionType("Exception message");
```

Finally Block: The finally block is an optional block that follows the catch block(s). It is executed regardless of whether an exception is thrown or not. It is commonly used for releasing resources or performing cleanup tasks.

```
try {  
  
    // Code that may throw an exception  
  
} catch (ExceptionType e) {  
  
    // Exception handling code  
  
} finally {  
  
    // Code to be executed regardless of exception occurrence  
  
}
```



The screenshot shows an IDE with a Java file named `Demo1.java` and a console window. The Java code is as follows:

```
1 package javaprograms;  
2  
3 //program to print the exception information using prints  
4  
5 import java.io.*;  
6  
7 class Demo1 {  
8     public static void main (String[] args) {  
9         int a=5;  
10        int b=0;  
11        try{  
12            System.out.println(a/b);  
13        }  
14        catch(ArithmeticException e){  
15            System.out.println("ArithmeticException");  
16        }  
17        finally{  
18            System.out.println("Hello");  
19        }  
20    }  
21 }  
22  
23  
24  
25  
26  
27
```

The console window shows the output of the program:

```
<terminated> Demo1 [Java Application] C:\Users\Akshatha\p2\pool\plugins\or  
ArithmeticException  
Hello
```

Checked vs Unchecked Exceptions: Java distinguishes between checked and unchecked exceptions. Checked exceptions are the exceptions that must be declared in the method signature or caught using try-catch blocks. Unchecked exceptions (such as RuntimeException) do not need to be declared or caught explicitly.

Exception Propagation: If an exception is not caught within a method, it is propagated up the call stack until it is caught or the program terminates. This allows exceptions to be handled in higher-level methods or in the main method.

Some of the types of exceptions are:

In Java, exceptions are categorized into two main types: checked exceptions and unchecked exceptions.

Checked Exceptions: Checked exceptions are the exceptions that must be declared in the method signature or caught using a try-catch block. These exceptions are typically used for conditions that can be reasonably anticipated and recovered from. They represent exceptional conditions that a well-written application should be able to handle.

Some examples of checked exceptions in Java include:

- **IO Exception:** Occurs when there is an input/output error.
- **SQL Exception:** Occurs when there is an error accessing a database.
- **ClassNotFoundException:** Occurs when a class is not found at runtime.
- **InterruptedException:** Occurs when a thread is interrupted while it is waiting, sleeping, or in a blocked state.

To handle a checked exception, you can either catch it using a try-catch block or declare it in the method signature using the throws keyword.

Unchecked Exceptions: Unchecked exceptions, also known as runtime exceptions, do not need to be declared or caught explicitly. They are typically caused by programming errors or unexpected conditions that are difficult to recover from. Unchecked exceptions do not require explicit handling because they are often caused by circumstances beyond the control of the programmer.

Some examples of unchecked exceptions in Java include:

- **NullPointerException:** Occurs when a null reference is accessed.
- **ArrayIndexOutOfBoundsException:** Occurs when an invalid index is used to access an array.

- **Arithmetic Exception:** Occurs when an arithmetic operation produces an exceptional condition, such as division by zero.
- **IllegalArgumentException:** Occurs when an illegal argument is passed to a method.

While it is not mandatory to handle unchecked exceptions, it is still recommended to handle them, when possible, to improve the robustness of your code. Unchecked exceptions can be caught and handled using a try-catch block, but it is more common to prevent them by writing defensive code and validating inputs.

Java also provides a hierarchy of exception classes, with Throwable as the root class. This hierarchy allows you to catch exceptions at different levels and handle them accordingly.

It's important to note that errors, such as OutOfMemoryError and StackOverflowError, are separate from exceptions. Errors indicate severe and unrecoverable conditions that are usually caused by problems in the JVM or the underlying system, and they are not meant to be caught or handled by the application code.

Multi-threading:

Multi-threading in Java allows you to execute multiple threads concurrently within a single program. A thread represents an independent unit of execution that can perform tasks simultaneously with other threads. Multi-threading is used to achieve concurrency, improve performance, and enhance the responsiveness of Java applications. Here are the key concepts and components related to multi-threading in Java:

Thread Class: The Thread class in Java provides the foundation for creating and managing threads. You can create a thread by extending the Thread class and overriding the run() method with the code that will be executed in the thread. The start() method is used to start the execution of the thread.

Runnable Interface: Alternatively, you can implement the Runnable interface and pass an instance of the implementing class to a Thread object. The run() method is defined in the Runnable interface and contains the code to be executed in the thread.

```
3 abstract class Shape {
4     String color;
5     abstract double area();
6     public abstract String toString();
7     public Shape(String color)
8     {
9         System.out.println("Shape constructor called");
10        this.color = color;
11    }
12    public String getColor() { return color; }
13 }
14 class Circle extends Shape {
15     double radius;
16
17     public Circle(String color, double radius)
18     {
19         super(color);
20         System.out.println("Circle constructor called");
21         this.radius = radius;
22     }
23
24     @Override double area()
25     {
26         return Math.PI * Math.pow(radius, 2);
27     }
28
29     @Override public String toString()
30     {
31         return "Circle color is " + super.getColor()
32             + " and area is : " + area();
33     }
34 }
35 class Rectangle extends Shape {
36     double length;
37     double width;
38
39     public Rectangle(String color, double length,
40                     double width)
41     {
42         // calling Shape constructor
43         super(color);
44         System.out.println("Rectangle constructor called");
45         this.length = length;
46         this.width = width;
```

```
<terminated> Demo1 [Java Application] C:\Users\Akshatha\p2\pool\plugins\o
Shape constructor called
Circle constructor called
Shape constructor called
Rectangle constructor called
Circle color is Red and area is : 15.205308443374602
Rectangle color is Yellow and area is : 8.0
```

Thread States: Threads can be in various states during their lifecycle, including:

- **New:** The thread has been created but has not yet started.
- **Runnable:** The thread is ready to run or is currently running.
- **Blocked:** The thread is waiting for a monitor lock to be released.
- **Waiting:** The thread is waiting indefinitely for another thread to perform a specific action.
- **Timed Waiting:** The thread is waiting for a specific period of time.
- **Terminated:** The thread has completed its execution and is terminated.

Thread Synchronization: When multiple threads access shared resources concurrently, synchronization is necessary to prevent race conditions and ensure data consistency. Java provides synchronized blocks and methods, as well as locks and conditions from the `java.util.concurrent` package, to achieve thread synchronization.

Thread Intercommunication: Threads can communicate and coordinate with each other using methods such as `wait()`, `notify()`, and `notifyAll()`. These methods are used in conjunction with the `synchronized` keyword to ensure proper inter-thread communication and synchronization.

Thread Priorities: Each thread in Java has a priority that determines its importance and scheduling preference. Thread priorities range from 1 (lowest) to 10 (highest). The `setPriority()` and `getPriority()` methods are used to set and retrieve the priority of a thread.

Thread Pooling: Creating and destroying threads can be resource-intensive. Thread pooling allows you to create a pool of reusable threads, managed by an executor, that can be assigned tasks as needed. This approach reduces the overhead of thread creation and provides better control over the number of concurrent threads

CHAPTER 5

CHALLENGES FACED

As a team member we had the opportunity to work on different languages and get along with the team mates.

As a manager we had to complete the pending work and assign the work to fellow team members based on their knowledge so that the time management of the project is on schedule

As a developer we have to come up with the creative method of writing the code and executing the developed code without errors

CHAPTER 6

CONCLUSIONS

During my internship, I had the opportunity to learn and work on various technologies, including JAVA, HTML, CSS, JS, and SQL. The internship program provided me with comprehensive training and hands-on experience in these programming languages and web technologies. Additionally, I had the privilege of working on a real-world project titled "Finding Voluntary Blood Donor."

Throughout the internship, I gained valuable knowledge and practical skills in software development, web design, and database management. I learned how to utilize JAVA for building robust applications, create dynamic and visually appealing web pages using HTML, CSS, and JS, and effectively manage data using SQL. The project on "Finding Voluntary Blood Donor" allowed me to apply my newly acquired skills to develop a functional system that addresses a critical social issue.

Through this internship, I not only enhanced my technical skills but also gained exposure to professional work environments, teamwork, and project management. The hands-on experience and guidance provided by the mentors and colleagues greatly contributed to my learning and personal growth.

100+ Years of Glorious history inscribed in the
yeomen service to the field of education

Centenary Celebrated Sharnbasveshwar Vidya Vardhak Sangha's

Estd. 2017
ಸ್ಥಾಪನೆ : 2017



Pooja Dr. Sharnbasappa Appaji
Mahatma Jyotiba Phule
Sharnbasveshwar Vidya Vardhak Sangha
President Sharnbasveshwar Vidya Vardhak Sangha
Chancellor, Sharnbasva University



Pooja Mahesh Dr. Dabkejini S. Appa
Chargeman,
Sharnbasveshwar Vidya Vardhak Sangha
Member of BOD, Sharnbasva University



Pooja Chiranjeevi Daddappa Appa
Mahatma Jyotiba Phule
Sharnbasveshwar Vidya Vardhak Sangha
Member of BOD, Sharnbasva University

ಶರಣಬಸವ
Sharnbasva



ವಿಶ್ವವಿದ್ಯಾಲಯ
University



Pooja Maheshri Godutai Arvaji



Pooja Daddappa Appa
Founder President
Sharnbasveshwar Vidya Vardhak Sangha

Kalaburagi - 585103, Karnataka - India

ಕಲಬುರಗಿ 585 103 ಕರ್ನಾಟಕ - ಭಾರತ

Phone / Fax No. 08472-277852, 277853, 277854, 277855 www.sharnbasvauniversity.edu.in - email : Sharnbasvauniversity@gmail.com

UGC Status: Letter No. F.8-29/2017(CPP-I/PU), Dated 20 Dec. 2017. Enlisted by the University Grants Commission, New Delhi, in the list of Private Universities in India.
A Private University enacted by Govt. of Karnataka as "Sharnbasva University Act. 2012" Karnataka Act No. 17 of 2013. Notification No. ED 144 URC 2016 dated 29/07/2017

STUDENT FEEDBACK OF INTERNSHIP

Student Name: Girish Patil

Date:

Industrial Supervisor: James

Title: MENTOR

Supervisor Email:

Internship is: [] Paid [X] Unpaid

Company/Organization: Ethnus Consultancy Services

Internship Address: No.151/17/1, SST Chambers, Second Floor, 36th Cross Road, Jayanagar 5th Block, Bengaluru 560041, India

Faculty Coordinator: Prof. Anil Dangi Department: Training & Placement

Dates of Internship: From 01-04-2023 to 10-06-2023

Please fill out the above in full detail

Give a brief description of your internship work (title and tasks for which you were responsible):

During my internship at Ethnus Consultancy Services, I was involved in various tasks related to web development and programming. My responsibilities included designing and developing web pages using HTML, CSS, and JavaScript, implementing client-side and server-side functionality, and working with databases using SQL. I also had the opportunity to work on a project titled "Finding Voluntary Blood Donor," where I developed a web application to connect blood donors with individuals in need. This project allowed me to apply my programming skills to address a social issue and contribute to the community.

Was your internship experience related to your major area of study?

[X] Yes, to a large degree

[] Yes, to a slight degree

[] No, not related at all

Indicate the degree to which you agree or disagree with the following statements:

This experience has:	Strongly Agree	Agree	No Opinion	Disagree	Strongly Disagree
Given me the opportunity to explore a career field	yes				
Allowed me to apply classroom theory to practice		yes			
Helped me develop my decision-making and problem-solving skills	yes				
Expanded my knowledge about the work world prior to permanent employment			yes		
Helped me develop my written and oral communication skills	yes				
Provided a chance to use leadership skills (influence others, develop ideas with others, stimulate decision-making and action)	yes				

This experience has:	Strongly Agree	Agree	No Opinion	Disagree	Strongly Disagree
Expanded my sensitivity to the ethical implications of the work involved		yes			
Made it possible for me to be more confident in new situations		yes			
Given me a chance to improve my interpersonal skills		yes			
Helped me learn to handle responsibility and use my time wisely			yes		
Helped me discover new aspects of myself that I didn't know existed before			yes		
Helped me develop new interests and abilities		yes			
Helped me clarify my career goals			yes		
Provided me with contacts which may lead to future employment			yes		

Allowed me to acquire information and/ or use equipment not available at my Institute		yes			
---	--	-----	--	--	--

In the Institute internship program, faculty members are expected to be mentors for students. Do you feel that your Faculty coordinator served such a function? Why or why not?

How well were you able to accomplish the initial goals, tasks and new skills that were set down in your learning contract? In what ways were you able to take anew direction or expand beyond your contract? Why were some goals not accomplished adequately?

In what areas did you most develop and improve?

What has been the most significant accomplishment or satisfying moment of your internship?

What did you dislike about the internship?

Considering your overall experience, how would you rate this internship? (Circle one).
(Satisfactory/ Good/ Excellent)

Give suggestions as to how your internship experience could have been improved. (Could you have handled added responsibility? Would you have liked more discussions with your professor concerning your internship? Was closer supervision needed? Was more of an orientation required

100+ Years of Glorious history inscribed in the
yeomen service to the field of education

Centenary Celebrated Sharnbasveshwar Vidya Vardhak Sangha's

Estd. 2017
ಸ್ಥಾಪನೆ : 2017



Pooja Dr. Sharnbasappa Appa
Kannada/History
Sharnbasveshwar Vidya Vardhak Sangha
President Sharnbasveshwar Vidya Vardhak Sangha
Channarayana, Sharnbasva University



Pooja Kalyani Dr. Subhagini S. Appa
Channarayana
Sharnbasveshwar Vidya Vardhak Sangha
Member of BOS Sharnbasva University



Pooja Channarayana Doddappa Appa
Kannada/History
Sharnbasveshwar Vidya Vardhak Sangha
Member of BOS Sharnbasva University

ಶರಣಬಸವ
Sharnbasva



ವಿಶ್ವವಿದ್ಯಾಲಯ
University



Pooja Matsuri Godutai Avaji
Sharnbasveshwar Vidya Vardhak Sangha



Pooja Doddappa Appa
Founder President
Sharnbasveshwar Vidya Vardhak Sangha

Kalaburagi - 585103, Karnataka - India

ಕಲಬುರಗಿ 585 103 ಕರ್ನಾಟಕ - ಭಾರತ

Phone / Fax No. 08472-277852, 277853, 277854, 277855 www.sharnbasvauniversity.edu.in - email : Sharnbasvauniversity@gmail.com

UGC Status: Letter No. F.8-29/2017(CPP-I/PU), Dated 20 Dec. 2017. Enlisted by the University Grants Commission, New Delhi, in the list of Private Universities in India.
A Private University enacted by Govt. of Karnataka as "Sharnbasva University Act. 2012" Karnataka Act No. 17 of 2013. Notification No. ED 144 URC 2016 dated 29/07/2017

STUDENT INTERNSHIP PROGRAM APPLICATION

1. Student Name:	Girish Patil		
2.Campus Address:	SHARNBASVA UNIVERSITY, KALABURAGI	Phone:9845852810	
3.Home Address:	NSL Sugars lts, tq-Aland, Dist-Kalburgi 585302	Phone:9901461251	
3a. Student email address:	gmp.kerur@gmail.com@gmail.com		
4.Academic Concentration	intern	5.Internship Semester: <u>4th</u> Year.	
6.Overall GPA:	NA		
Location		Core Area	Company/ institution
Preference-1	offline	Jayanagar , Banglore	institution
Preference-2			
Preference-3			
Faculty mentor Signature:___Date__.			
Signature confirms that the student has attended the internship orientation and has met all paperwork and process requirements to participate in the internship program,and has received approval from his/her Advisor..			
Student Signature:___Date__.			
Signature confirms that the student agrees to the terms, conditions, and requirements of the Internship Program			

100+ Years of Glorious history inscribed in the
yeomen service to the field of education

Centenary Celebrated Sharnbasveshwar Vidya Vardhak Sangha's

Estd. 2017
ಸ್ಥಾಪನೆ : 2017



Pooja Dr. Sharnbasappa Appaji
Maharishi Prathodigala
Sharnbasveshwar Vidya Vardhak Sangha
Chancellor, Sharnbasva University



Pooja Maheshwari Dr. Sharnbasappa S. Appaji
Chancellor
Sharnbasveshwar Vidya Vardhak Sangha
Member of BOD, Sharnbasva University



Pooja Chittambur Sharnbasappa Appaji
Maharishi Prathodigala
Sharnbasveshwar Vidya Vardhak Sangha
Member of BOD, Sharnbasva University

ಶರಣಬಸವ
Sharnbasva



ವಿಶ್ವವಿದ್ಯಾಲಯ
University



Pooja Maheshwari Goduti Anjali



Pooja Doodappa Appa
Founder President
Sharnbasveshwar Vidya Vardhak Sangha

Kalaburagi - 585103, Karnataka - India

ಕಲಬುರಗಿ 585 103 ಕರ್ನಾಟಕ - ಭಾರತ

Phone / Fax No. 08472-277852, 277853, 277854, 277855 www.sharnbasvauniversity.edu.in - email : Sharnbasvauniversity@gmail.com

UGC Status: Letter No. F.8-29/2017(CPP-I/PU), Dated 20 Dec. 2017. Enlisted by the University Grants Commission, New Delhi, in the list of Private Universities in India.
A Private University enacted by Govt. of Karnataka as "Sharnbasva University Act. 2012" Karnataka Act No. 17 of 2013. Notification No. ED 144 URC 2016 dated 29/07/2017

OBJECTIVES/ GUIDELINES/ AGREEMENT: INTERNSHIP SYNOPSIS (THIS WILL BE PREPARED IN CONSULTATION WITH FACULTY MENTOR)

An internship is a unique learning experience that integrates studies with practical work. This agreement is written by the student in consultation with the faculty Mentor and Industrial supervisor. It shall serve to clarify the educational purpose of the internship and to ensure an understanding of the total learning experience among the principal parties involved.

Part I: Contact Information Student

Name: Girish Patil Student ID SG19CSE053 Class Year: Final year
Campus Address: Vidyanagar
City, State: Kalaburagi, Karnataka
Phone: 9845852810 Email: gmp.kerur@gmail.com

Industrial Supervisor

Name: Title: _____ Company/Organization: _____
Internship Address: _____ City: _____ State: _____ Pin: _____
Phone: _____ Email: _____

Faculty Mentor

Name: Prof. Anil Dangi Phone: 9739674760
Campus Address: vidya nagar, kalaburagi

Academic Credit Information

Internship Title: _____ Department: _____
Course #: _____ Credits: _____
Grading Option: _____ Credit/Non-credit _____
Beginning Date: _____ Ending Date: _____
Hours per Week: _____ Internship is: _____ Paid _____ Unpaid _____

100+ Years of Glorious history inscribed in the
yeomen service to the field of education

Centenary Celebrated Sharnbasveshwar Vidya Vardhak Sangha's

Estd. 2017
ಸ್ಥಾಪನೆ : 2017



ಶರಣಬಸವ
Sharnbasva



ವಿಶ್ವವಿದ್ಯಾಲಯ
University



Kalaburagi - 585103, Karnataka - India

ಕಲಬುರಗಿ 585 103 ಕರ್ನಾಟಕ - ಭಾರತ

Phone / Fax No. 08472-277852, 277853, 277854, 277855 www.sharnbasvauniversity.edu.in - email : Sharnbasvauniversity@gmail.com

UGC Status: Letter No. F.8-29/2017(CPP-I/PU), Dated 20 Dec. 2017. Enlisted by the University Grants Commission, New Delhi, in the list of Private Universities in India.
A Private University enacted by Govt. of Karnataka as "Sharnbasva University Act. 2012" Karnataka Act No. 17 of 2013. Notification No. ED 144 URC 2016 dated 29/07/2017

Part II: Internship Objectives/Learning Activities

Internship Objectives: What do you intend to learn, acquire, and clarify through this internship? Try to use concrete, Measurable terms in listing your learning objectives under each of the following categories:

- Knowledge and Understanding
- Skills

Learning Activities: How will your internship activities enable you to acquire the knowledge/understanding, and skills you listed above.

On the job: Describe how your internship activities will enable you to meet your learning objectives. Include projects, research, report writing, conversations, etc., which you will do while working, relating them to what you intend to learn.

Teaching/Mentoring Activities: How your technical knowledge can be applied at the site of the internship. How you can create value through mentoring/help people learn new things.

Off the job: List reading, writing, contact with faculty supervisor, peer group discussion, field trips, observations, etc., you will make and carry out which will help you meet your learning objectives.

Evaluation: Your Internship supervisor will provide a written evaluation of your internship. Describe in detail what other evidence you will provide to your faculty Mentor to document what you have learned (e.g. journal, analytic paper, project, descriptive paper, oral presentation, etc.) Include deadline dates.

Part III: The Internship

Job Description: Describe in as much detail as possible your role and responsibilities while on your internship. List Duties, project to be completed, deadlines, etc. How can you contribute to the organization/site of internship?

Supervision: Describe in as much detail as possible the supervision to be provided/needed at the work site. List What kind of instruction, assistance, and consultation you will receive from whom, etc.

Evaluation: How will your work performance be evaluated? By whom? When?

Part IV: Agreement

This contract may be terminated or amended by student, faculty coordinator or work supervisor at any time upon Written notice, which is received and agreed to by the other two parties.

Student: Girish Patil

Date

Faculty Mentor: Prof. Anil Dangi

Date

Industry Supervisor S Radha

Date

***Photocopy of the attendance record duly attested by the training in-charge should be attached with the evaluation Performa.**



INTERNSHIP EVALUATION REPORT

(For UG/ PG COURSES)

Name & Address of Organization

Ethus Consultancy Services

No.151/17/1, SST Chambers, Second Floor, 36th Cross Road, Jayanagar
5th Block Bengaluru 560041. India

Sr. No.	Name of Student	Roll No.	Marks to be awarded by			OVERALL GRADE
			Punctua lity Grade (Satisfa ctory/ Good/ Excele nt)	Maintenan ce of Daily Diary Grade (Satisfactor y/ Good/ Excellent)	Skill Test Grade (Satisfactory / Good/ Excellent)	
1	Girish Patil	SG19CSE053				

Excellent – A+ Good – A Satisfactory - B

100+ Years of Glorious history inscribed in the
yeomen service to the field of education

Centenary Celebrated Sharnbasveshwar Vidya Vardhak Sangha's

Estd. 2017
ಸ್ಥಾಪನೆ : 2017



Pooja Dr. Sharnbasappa Appaji
Karnataka Teachers' Guild
Sharnbasveshwar Vidya Vardhak Sangha
President of Sharnbasveshwar Vidya Vardhak Sangha
Dharwad, Sharnbasva University



Pooja Matoshi Dr. Dadasaheb S. Appa
Chairperson
Sharnbasveshwar Vidya Vardhak Sangha
Member of UGC, Sharnbasva University



Pooja Chiranjeevi Daddappa Appa
Mahadevi 18 Prathibadipya
Sharnbasveshwar Vidya Vardhak Sangha
Dharwad, Sharnbasva University

ಶರಣಬಸವ
Sharnbasva



ವಿಶ್ವವಿದ್ಯಾಲಯ
University



Pooja Matoshi Godutal Avaji



Pooja Daddappa Appa
Founder-President
Sharnbasveshwar Vidya Vardhak Sangha

Kalaburagi - 585103, Karnataka - India

ಕಲಬುರಗಿ 585 103 ಕರ್ನಾಟಕ - ಭಾರತ

Phone / Fax No. 08472-277852, 277853, 277854, 277855 www.sharnbasvauniversity.edu.in - email : Sharnbasvauniversity@gmail.com

UGC Status: Letter No. F.8-29/2017(CPP-I/PU), Dated 20 Dec. 2017. Enlisted by the University Grants Commission, New Delhi, in the list of Private Universities in India.
A Private University enacted by Govt. of Karnataka as "Sharnbasva University Act. 2012" Karnataka Act No. 17 of 2013. Notification No. ED 144 URC 2016 dated 29/07/2017

ATTENDANCE SHEET

(For UG/ PG COURSES)

Name & Address of Organization

Ethus Consultancy Services

No.151/17/1, SST Chambers, Second Floor, 36th Cross Road, Jayanagar

5th Block Bengaluru 560041. India

Name of Student	Girish Patil	
Roll. No	SG19CSE053	
Name of Course	Computer Science Engineering	
Date of Commencement of Training.:	01 Apr 2023	
Date of Completion of Training:	25 June 2023	

Initials of the student

Mo nth & Yea r	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31

Note:

1. Attendance Sheet should remain affixed in Daily Training Diary.
2. Do not remove or tear it off.
3. Student should sign/initial in the attendance column. Do not mark 'P'
4. Holidays should be marked in **Red Ink** in attendance column.
5. Absent should be marked as 'A' in Red Ink.

(Name _____) Contact

Signature of Company
Internship supervisor.