DEPARTMENT OF EDUCATION
REGION X - NORTHERN MINDANAO
DIVISION OF CAGAYAN DE ORO CITY
Fr. William F. Masterson, SJ Avenue, Upper Balulang, Cagayan de Oro City

Learning Activity Sheets

JAVA Programming

SHARED OPTIONS
Senior High Alternative Responsive Education Delivery

Preface

It has been elaborated in research and literature that the highest performing education systems are those that combine quality with equity. Quality education in the Department of Education (DepEd) is ensured by the learning standards in content and performance laid in the curriculum guide. Equity in education means that personal or social circumstances such as gender, ethnic origin or family background, are not obstacles to achieving educational potential and that inclusively, all individuals reach at least a basic minimum level of skills.

In these education systems, the vast majority of learners have the opportunity to attain high-level skills, regardless of their own personal and socio-economic circumstances. This corresponds to the aim of DepEd Cagayan de Oro City that no learner is left in the progression of learning. Through DepEd's flexible learning options (FLO), learners who have sought to continue their learning can still pursue in the Open High School Program (OHSP) or in the Alternative Learning System (ALS).

One of the most efficient educational strategies carried out by DepEd Cagayan de Oro City at the present is the investment in FLO all the way up to senior high school. Hence, Senior High School Alternative Responsive Education Delivery (SHARED) Options.

Two secondary schools, Bulua National High School and Lapasan National High School, and two government facilities, Bureau of Jail Management and Penology-Cagayan de Oro City Jail and Department of Health-Treatment and Rehabilitation Center-Cagayan de Oro City, are implementing the SHARED Options.

To keep up with the student-centeredness of the K to 12 Basic Education Curriculum, SHARED Options facilitators are adopting the tenets of Dynamic Learning Program (DLP) that encourages responsible and accountable learning.

This compilation of DLP learning activity sheets is an instrument to achieve quality and equity in educating our learners in the second wind. This is a green light for SHARED Options and the DLP learning activity sheets will continually improve over the years.

Ray Butch D. Mahinay, PhD
Jean S. Macasero, PhD

Acknowledgment

The operation of the Senior High School Alternative Responsive Education Delivery (SHARED) Options took off with confidence that learners with limited opportunities to senior high school education can still pursue and complete it. With a pool of competent, dedicated, and optimistic Dynamic Learning Program (DLP) writers, validators, and consultants in Senior High School Technical Vocational Livelihood Learning activity Sheets , the SHARED Options is in full swing.

Gratitude is due to the following:

- Schools Division Superintendent, Cherry Mae L. Limbaco, PhD, CESO V, Assistant Schools Division Superintendent Alicia E. Anghay, PhD, for buoying up this initiative to the fullest;
- CID Chief Lorebina C. Carrasco, and SGOD Chief Rosalio R. Vitorillo, for the consistent support to all activities in the SHARED Options;
- School principals and senior high school teachers from Bulua NHS, Lapasan NHS, Puerto NHS and Lumbia NHS, for the legwork that SHARED Options is always in vigor;
- Stakeholders who partnered in the launching and operation of SHARED Options, specifically to the Bureau of Jail Management and Penology-Cagayan de Oro City Jail and the Department of Health-Treatment and Rehabilitation Center-Cagayan de Oro City;
- Writers Maryann S. Macaron and validators of the DLP learning activity sheets, to which this compilation is heavily attributable to, for their expertise and time spent in the workshops;

- Alternative Learning System implementers namely Willy P. Calo Ailiene P. Libres, Rubeneth V. Salazar and Metocila O. Agbay, Puerto National High School, Leneth G. Udarbe, Lapasan National High School and Pinky B. Dela Calzada, for the technical assistance given to the sessions;
- Reproduction (LRMDS) Gemma P. Pajayon and Lanie M. Signo;
- To all who in one way or another have contributed to the undertakings of SHARED Options.

Mabuhay ang mga mag-aaral! Ito ay para sa kanila, para sa bayan!

Jean S. Macasero, PhD
Juan A. Mingo

TABLE OF CONTENTS

Computer Programming in Java (NC III)

34	Java methods with arguments and return values			15
35	Methods with arguments and return values			20
36	Static keywords and its applications			10
37	Static keywords and its applications: Static Variables			15
38	Method Overloading			25
39	Method Overloading: Changing Data Type of Arguments			25
40	Read and Write Data from the Console			10
41	Read and Write Data from the Console			40
42	Java Files			10
43	Java Files: Create a File			20
44	Java Files: Write to a File			20
45	Java Files: Read a File			20
46	Java Access Modifier: Default Access Modifier			10
47	Java Access Modifier: Private Access Modifier			10
48	Java Access Modifier: Public Access Modifier			20
49	Java Access Modifier: Protected Access Modifier			5
50	Java Type Casting: Widening			10
51	Java Type Casting: Narrowing			10
52	Java Virtual Methods			15
53	Method Overriding			10
54	Method Overriding: A Program			20
55	Package and import statements and its uses			10
56	Package and import statements and its uses: A Program			20
57	Java Abstract Classes: Its Uses			10
58	Static and Final Keywords			10
59	Nested Classes			15
60	Using enumerated types			15
61	Interfaces			10
62	Implement Interfaces			50
63	Java Generics			10
64	Using throw statements			15
65	JDBC API			15
66	Using JDBC driver to connect to database			10
67	Using JDBC driver to connect to database			30
68	Applying JDBC Row Set Provider, Row Set Factory, and Row Set Interfaces			30
69	Creating and using PreparedStatement			10
70	Creating and using PreparedStatement- Insert Record			30
71	Creating and using PreparedStatement- Update Record			30
72	Creating and using PreparedStatement- Delete Record			30

DepED DEPARTMENT OF EDUCATION
K TO 12 PROGRAM

SHARED OPTIONS
SENIOR HIGH ALTERNATIVE RESPONSIVE EDUCATION DELIVERY
GRADE 11 DLP LEARNING ACTIVITY SHEET

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Java technology and the Java programming language		
Lesson Competency : Identify knowledge of Java technology and Java programming		
References:https://docs.oracle.com/javase/tutorial/getStarted/intro/definition.html		LAS No.: 1

CONCEPT NOTES

Java Technology is both a programming language and a platform.

The Java Programming Language is a high-level language characterized by the following:

- **Simple** - can be programmed without extensive programmer training
- **Object-oriented** - provides a clean and efficient object-based development platform.
- **Distributed**- designed to operate in distributed environments
- **Interpreted** - The *Java interpreter* can execute Java bytecodes directly on any machine to which the interpreter and run-time system have been ported.
- **Robust** - It provides extensive compile-time checking.
- **Secure** - Java technology lets you construct applications that can't be invaded from outside.
- **Architecture neutral** - transport code efficiently to multiple hardware and software platforms.
- **Portable** - Your programs are the same on every platform
- **High performance** - adopt a scheme by which the interpreter can run at full speed without needing to check the run-time environment.
- **Multithreaded** - provides the means to build applications with many concurrent threads of activity.
- **Dynamic** - Classes are linked only as needed.

EXERCISES

Identification: Identify what is being described by the following statements. Write your answer on the space provided before each number. (2 points each)

_____1. Java can be easily programmed by most developers without extensive training.

_____2. Java can perform multiple concurrent activities.

_____3 Programs are the same on every platform.

_____4. Java technology lets you construct applications that can't be invaded from outside.

_____5. Is both a programming language and a platform.

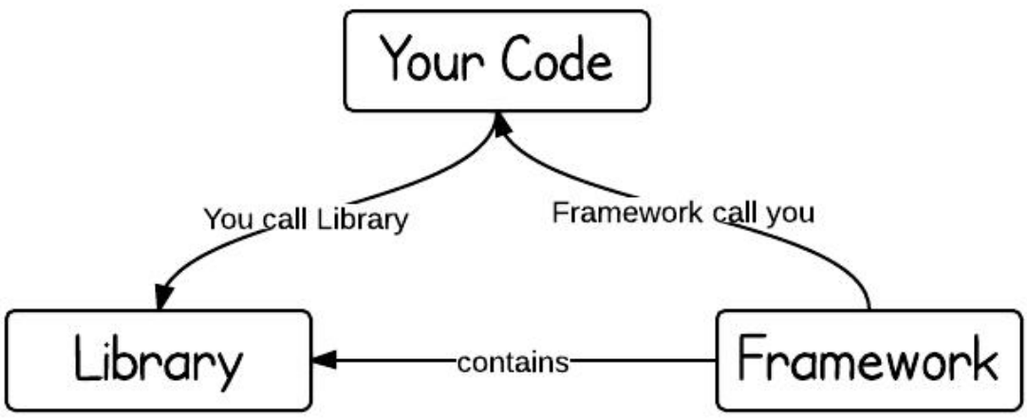
Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Java Framework		
Lesson Competency : Differentiate Java Framework from Java Library		
References :http://www.fredosaurus.com/notes-java/oodesign/frameworks.html		LAS No.: 2

CONCEPT NOTES

Java Frameworks are large bodies (usually many classes) of prewritten code to which you add your own code to solve a problem in a specific domain.

Java Framework vs Java Library

The key difference between a library and a framework is "Inversion of Control". When you call a method from a library, you are in control. But with a framework, the control is inverted: the framework calls you.



Usefulness of Frameworks

A framework will often dictate the structure of your application. Some frameworks even supply so much code that you have to do very little to write your application. This can be good or bad, depending on how easy it is to use

Examples in Java

-Swing classes
- AWT classes

EXERCISES

Discussion. Discuss the following:

1. Differentiate Java framework from Java library in your own words. (10 points)

DepED DEPARTMENT OF EDUCATION
K TO 12 PROGRAM

SHARED OPTIONS
SENIOR HIGH ALTERNATIVE RESPONSIVE EDUCATION DELIVERY
GRADE 11 DLP LEARNING ACTIVITY SHEET

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Importing Java Packages		
Lesson Competency : Import Java packages to make them accessible in the code		
References : https://www.studytonight.com/java/package-in-java.php		LAS No.: 3

CONCEPT NOTES

Java Package

- are used in Java, in-order to avoid name conflicts and to control access of class, interface and enumeration etc. A package can be defined as a group of similar types of classes, interface, enumeration or sub-package.

How to create a package in Java?

Simply include a package command followed by name of the package as the first statement in java source file.

```
package sample;
public class Employee{
        statement;
}
```

How to import a package?

import keyword is used to import built-in and user-defined packages into your java source file so that your class can refer to a class that is in another package by directly using its name.

Example:

```
package sample;
import java.util.Scanner;
public class Employee{
        statement;
}
```

EXERCISES.

Writing Codes. Perform the following:

Given the following codes in Java:

```
package trial;
public class TryMe{
}
```

1. Re-write the codes such that the package **javax.swing.JOptionPane** will be imported to your program. (5 points)

DepED DEPARTMENT OF EDUCATION
K TO 12 PROGRAM

SHARED OPTIONS
SENIOR HIGH ALTERNATIVE RESPONSIVE EDUCATION DELIVERY
GRADE 11 DLP LEARNING ACTIVITY SHEET

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Java Identifiers		
Lesson Competency : Identify and create valid Java identifiers		
References : Malik, D. (2012). *Java programming.* [Boston, Mass.]: Course Technology Cengage Learning, p.31.		LAS No.: 4

CONCEPT NOTES

Identifiers refers to names of things, such as variables, constants, and methods that appear in programs. Some identifiers are predefined; others are defined by the user. All identifiers must obey Java's rules for identifiers.

Rules in defining identifier:

- A Java identifier consists of letters, digits, the underscore character (_), and the dollar sign ($) and must begin with a letter, underscore, or the dollar sign.
- Identifiers can be made of only letters, digits, the underscore character (_), and the dollar sign ($); no other symbols are permitted to form an identifier.
- Identifiers can be of any length.

The following are legal identifiers in Java:

first_Name	counter1
number	&Total
payRate	_Amount

EXERCISES

1. Which of the following are valid identifiers? Mark the following as valid or invalid and write your answer on the space provided before each number.

_____a. myFirstProgram	_____f. 1footEquals12Inches
_____b. MAKE-UP	_____g. Yan'sFirstTry
_____c. JavaProgram2	_____h. Save Amount
_____d. score1	_____i. 6th
_____e. Programming2Lecture	_____j. New_Enrollee

2. Write at least 5 valid identifiers. (5 points)

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : JAVA data types		
Lesson Competency : Identify appropriate Java Data types in accordance with Java framework		
References : https://www.javatpoint.com/java-data-types		LAS No.: 5

CONCEPT NOTES

Java Data Types

Data types specify the different sizes and values that can be stored in the variable. There are two types of data types in Java:

1. **Primitive data types:** The primitive data types include boolean, char, byte, short, int, long, float and double.
2. **Non-primitive data types:** The non-primitive data types include Classes, Interfaces, and Arrays.

Data Type	Default Value	Default Size
boolean	**false**	**1 bit**
char	'\u0000'	2 byte
byte	**0**	**1 byte**
short	0	2 byte
int	0	4 byte
long	0L	8 byte
float	0.0f	4 byte
double	0.0d	8 byte

Sample Usage

Problem: Total amount of purchased items.

Appropriate data type: double, float or long because these data types can numbers with decimal places.

Declaration of variables data type in Java:

```
int number;         // declare variable number of type int
char letter ='a';  // declare variable letter of type char and assigned a value 'a'
```

EXERCISES. Identification. Identify the appropriate data type for the following statements. Write your answer on the space provided before each number. (2 points each)

_____1. Computed average grade.
_____2. Multiple choice options like a, b and c's.
_____3. A collection of Peso coins.
_____4. Number of students in the classroom.
_____5. Length of the green board.

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Operators in Java		
Lesson Competency : Identify operators used in Java programming		
References : https://www.tutorialspoint.com/java/java_basic_operators		LAS No.: 6

CONCEPT NOTES

Operators in Java

Java provides a rich set of operators to manipulate variables.

The following are the commonly used operators in Java:

- Arithmetic Operators
- Relational Operators
- Logical Operators

Arithmetic Operators- can be applied on any numeric type: byte, short, int, long, float, or double.

- Addition (+), Subtraction (-), Multiplication (*), Division (/), Modulus (%), Increment (++), Decrement (--)

Relational Operators- determines the relationship between two operands

- Equal to (==), Not equal to (!=), Greater than (>), Greater than or Equal to (>=), Less than (<), Less than or equal to (<=)

Logical Operators- used when we want to check multiple conditions together.

- Logical And (&&), Logical Or (||), Logical Not (!)

EXERCISES

Identification. Answer the following questions. Write your answer on the space provided before each number. (2 points each)

_____1. These operators are used to perform mathematical operations.

_____2. How many operands are required when using the relational operators?

_____3. These operators are used in checking multiple conditions together.

_____4. What kind of operator is increment?

_____5. What is being manipulated when we use Java operators?

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Operators in Java: Arithmetic Operators		
Lesson Competency : Evaluate expressions using Arithmetic Operators in Java		
References : https://www.tutorialspoint.com/java/java_basic_operators		LAS No.: 7

CONCEPT NOTES

The Arithmetic Operators are used in mathematical expressions in the same way that they are used in algebra. The following table lists the arithmetic operators:

Assume integer variable A holds 10 and variable B holds 20, then –

Operator	Description	Example
+ (Addition)	Adds values on either side of the operator.	A + B will give 30
- (Subtraction)	Subtracts right-hand operand from left-hand operand.	A - B will give -10
* (Multiplication)	Multiplies values on either side of the operator.	A * B will give 200
/ (Division)	Divides left-hand operand by right-hand operand.	B / A will give 2
% (Modulus)	Divides left-hand operand by right-hand operand and returns remainder.	B % A will give 0
++ (Increment)	Increases the value of operand by 1.	B++ gives 21
-- (Decrement)	Decreases the value of operand by 1.	B-- gives 19

Arithmetic Expression	Result	Description
5 / 2	2	In the division 5 / 2, the quotient is 2 and the remainder is 1. Therefore, 5 / 2 with the integral operands evaluates to the quotient, which is 2.
34 % 5	4	In the division 34 / 5, the quotient is 6 and the remainder is 4. Therefore, 34 % 5 evaluates to the remainder, which is 4.
5/2.0	2.5	5 is of type int, 2.0 is double making the result double.

EXERCISES

Evaluation. Evaluate the following expressions. Write your answer on the space provided after each number. (2 points each)

a. 25 / 3 = _____
b. 20 - 12 / 4 * 2 = _____
c. 32 % 7 = _____
d. 3 - 5 % 7 = _____
e. 18.0 / 4 = _____
f. 28 - 5 / 2.0 = _____
g. 17 + 5 % 2 – 3 = _____
h. 15.0 + 3.0 * 2.0 / 5.0 = _____
i. 17 % 3 + 4 / 3 = _____
j. 15.0 / 2 * 4 = _____

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Operators in Java: Relational Operators		
Lesson Competency : Evaluate expressions using Relational Operators in Java		
References : https://www.tutorialspoint.com/java/java_basic_operators		LAS No.: 8

CONCEPT NOTES

The Relational Operators

The following are the relational operators supported by Java language.

Assume variable A holds 10 and variable B holds 20, then –

Operator	Description	Example
== (equal to)	Checks if the values of two operands are equal or not, if yes then condition becomes true.	(A == B) is not true.
!= (not equal to)	Checks if the values of two operands are equal or not, if values are not equal then condition becomes true.	(A != B) is true.
> (greater than)	Checks if the value of left operand is greater than the value of right operand, if yes then condition becomes true.	(A > B) is not true.
< (less than)	Checks if the value of left operand is less than the value of right operand, if yes then condition becomes true.	(A < B) is true.
>= (greater than or equal to)	Checks if the value of left operand is greater than or equal to the value of right operand, if yes then condition becomes true.	(A >= B) is not true.
<= (less than or equal to)	Checks if the value of left operand is less than or equal to the value of right operand, if yes then condition becomes true.	(A <= B) is true.

EXERCISES. Evaluation. Determine whether the following expressions evaluates to true or false. (2 points each)

_____1. 12 > 7
_____2. (6 % 4) < 8
_____3. (2 * 4) >= (5 + 3)
_____4. (12 / 4) != (5 - 2)
_____5. (6 * 2 - 4) == (5 + 1 * 2)
_____6. (10 – 2) <= (5 + 3)
_____7. 8 == (4 * 3)
_____8. (15 % 3) < (5 - 2)
_____9. (9/3 + 2)==(15 % 4 + 1)
_____10. (6 * 3 % 4) > 7

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Operators in Java: Logical Operators		
Lesson Competency : Evaluate expression using Logical Operators in Java		
References : https://www.tutorialspoint.com/java/java_basic_operators		LAS No.: 9

CONCEPT NOTES

The Logical Operators

The following table lists the logical operators:

Assume Boolean variables A holds true and variable B holds false, then,

Operator	Description	Example
&& (logical and)	Called Logical AND operator. If both the operands are non-zero, then the condition becomes true.	(A && B) is false
|| (logical or)	Called Logical OR Operator. If any of the two operands are non-zero, then the condition becomes true.	(A || B) is true
! (logical not)	Called Logical NOT Operator. Use to reverses the logical state of its operand. If a condition is true then Logical NOT operator will make false.	!(A && B) is true

Example:

(5 > 8) && (3 < 5) is **false** because 5 > 8 is false and 3 is less than 5 is true. Therefore, false and true will result to false base on the above table.

EXERCISES. Evaluation. Suppose that x, y, and z are int variables and x = 5, y = 10, and z = 15. Determine whether the following expressions evaluates to true or false. Write your answer on the space provided before each number. (2 points each)

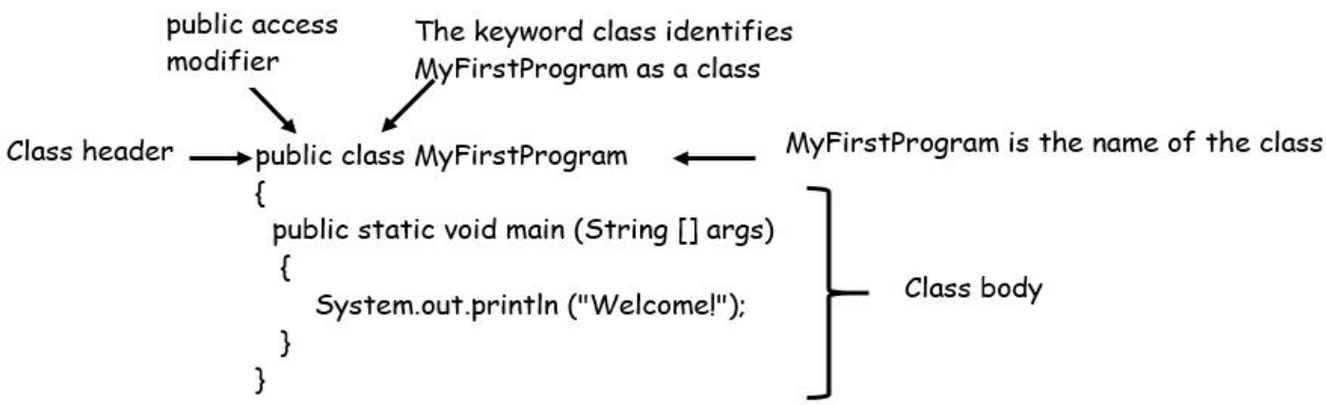
_____1. ! (x > 5)
_____2. x <= 5 || y < 15
_____3. (x != 5) && (y != z)
_____4. x >= z || (x + y >= z)
_____5. (x <= y – 2) && (y >= z) || (z – 2 != 20)

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Define the Structure of Java Class		
Lesson Competency : Create basic Java Class		
References : http://ecomputernotes.com/java/what-is-java/helloworldjavaexample		LAS No.: 10

CONCEPT NOTES

Class is a blueprint or a set of instruction to build a specific type of object.

Parts of Java



1. Definition of the class - The simplest class definition consists of the keyword class, followed by the class name. In our case, the class is **MyFirstProgram**.
2. Definition of the method main () - a gateway or starting point for the program.
3. Content of the method main () – actual coding starts

In the image above, the program will simply prints **Welcome!** in the screen using System.out.println(). Anything that is inside the double quotes will be printed.

Sample Program:

```
public class ASimpleJavaProgram
{
	public static void main(String[] args)
	{	System.out.println("My first Java program.");
		System.out.println("The sum of 2 and 3 = " + 5);
		System.out.println("7 + 8 = " + (7 + 8));
}}
```

When compiled, the following will be printed:

```
My first Java program.
The sum of 2 and 3 = 5
7 + 8 = 15
```

EXERCISES

Performance Task. Perform the following. (10 points)

Create a program that prints your Full Name and your school name on the screen. Name your program as **MyProgram.java.**

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Selection Statement		
Lesson Competency : Identify the types of selection statements in accordance with Java framework		
References : Pearson.com. (2019). *Liang, Intro to Java Programming, Comprehensive Version	Pearson*. [online] Available at: https://www.pearson.com/us/higher-education/product/Liang-Intro-to-Java-Programming-Comprehensive-Version-10th-Edition/9780133761313.html [Accessed 18 Jul. 2019].		LAS No.: 11

CONCEPT NOTES

Selection statements use conditions that are Boolean expressions. *A Boolean expression* is an expression that evaluates to a *Boolean value*: **true** or **false** using relational operators.

Types of Selection Statements

1. One way if Statement - executes an action if and only if the condition is **true**.
2. Two-way if-else Statements - decides the execution path based on whether the condition is **true** or **false**.
3. Nested if Statements - An **if** statement can be inside another **if** statement to form a nested **if** statement.
4. Switch statements - based on the value of a variable or an expression.
    - Simplify coding for multiple conditions.

EXERCISES

Identification. Identify the following. Write your answer on the space provided before each number. (2 points each)

_____1. What do you call the true and false values?
_____2. A type of selection statement that yields no output if the condition is false.
_____3. It simplifies coding for multiple conditions.
_____4. An if statement inside an if statement.
_____5. A construct that enables a program to specify alternative paths of execution.

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Selection Statement: If Statement		
Lesson Competency : Create program that uses if statement in accordance with Java framework		
References : Pearson.com. (2019). *Liang, Intro to Java Programming, Comprehensive Version	Pearson.* [online] Available at: https://www.pearson.com/us/higher-education/product/Liang-Intro-to-Java-Programming-Comprehensive-Version-10th-Edition/9780133761313.html [Accessed 18 Jul. 2019].		LAS No.: 12

CONCEPT NOTES

If Statement is a construct that enables a program to specify alternative paths of execution.

The syntax for a one-way **if** statement is:

```
if (boolean-expression) {
        statement(s);
}
```

Sample Program: A program that prints "Congratulations! You passed" if the grade is greater than or equal to 75.

Solution:

```
public class SimpleIfDemo {
    public static void main(String[] args) {
        int grade = 90;

        if (grade>=75)
            System.out.println("Congratulation! You passed.");
    }
}
```

Output:

Congratulations! You passed.

EXERCISES.

Performance task. Perform the following. (15 points)

Write a program that declares a variable named amount of type double and set its value to 1500. If amount is above 1000, you program will print "You have sufficient amount in your account." Save your program as **MyIfProgram.java.**

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Selection Statement: Two-Way If-Else Statement		
Lesson Competency : Create program that uses two-way if-else statement in accordance with Java framework		
References : Pearson.com. (2019). *Liang, Intro to Java Programming, Comprehensive Version	Pearson*. [online] Available at: https://www.pearson.com/us/higher-education/product/Liang-Intro-to-Java-Programming-Comprehensive-Version-10th-Edition/9780133761313.html [Accessed 18 Jul. 2019].		LAS No.: 13

CONCEPT NOTES

Two-Way **if-else** Statements

The actions that a two-way **if-else** statement specifies differ based on whether the condition is **true** or **false**.

Syntax:

```
if (boolean-expression) {
    statement(s)-for-the-true-case;
}
else {
    statement(s)-for-the-false-case;
}
```

Sample Program: A program that prints "Congratulations! You passed" if the grade is greater than or equal to 75 otherwise, prints "Sorry, you failed."

Solution:

```
public class SimpleIfDemo {
    public static void main(String[] args) {
        int grade = 70;
        if (grade>=75)
            System.out.println("Congratulation! You passed.");
        else
            System.out.println("Sorry, you failed.");
    }
}
```

Output: **Sorry, you failed.**

EXERCISES. Performance Task. Perform the following: (15 points)

Write a program that declares a variable named amount of type double and set its value to 800. If amount is above 1000, you program will print "You have sufficient amount in your account" else print "Sorry, you don't have enough balance in your account." Save your program as **MyIfProgram.java.**

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Selection Statement: Nested If and Multi-Way If-Else Statements		
Lesson Competency : Identify the output of the program that uses nested if and multi-way if-else statements in accordance with Java framework		
References : Pearson.com. (2019). *Liang, Intro to Java Programming, Comprehensive Version	Pearson.* [online] Available at: https://www.pearson.com/us/higher-education/product/Liang-Intro-to-Java-Programming-Comprehensive-Version-10th-Edition/9780133761313.html [Accessed 18 Jul. 2019].	LAS No.: 14	

CONCEPT NOTES

Nested If and Multi-Way If-Else Statements

An **if** *statement can be inside another* **if** *statement to form a nested* **if** *statement.* The statement in an **if** or **if-else** statement can be any legal Java statement, including another **if** or **if-else** statement. The inner **if** statement is said to be *nested* inside the outer **if** statement. The following is a nested **if** statement:

```
if (i > k) {
    if (j > k)
    System.out.println("i and j are greater than k"); }
else
    System.out.println("i is less than or equal to k");
```

A preferred format for multiple alternatives is shown in (b).

```
if(score>=90.0)
    System.out.print("A");
else
    if (score>=80.0)
        System.out.print("B");
    else
        if(score>=70.0)
            System.out.print("C");
        else
                        System.ou
```

(a)

Equivalent

(b)

```
if(score>=90.0)
    System.out.print("A");
else  if (score>=80.0)
    System.out.print("B");
else if(score>=70.0)
    System.out.print("C");
else
    System.out.print("F");
```

EXERCISES. Program Tracing. Suppose x = 3 and y = 2; show the output, if any, of the following code segment. (5 points each)

```
if (x > 2) {
    if (y > 2) {
        z = x + y;
        System.out.println("z is " + z);}
    }
    else
        System.out.println("x is " + x);
```

Write your output here:

a. What is the output if x = 3 and y = 4? _____

b. What is the output if x = 2 and y = 2? _____

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Selection Statement: Switch Statements		
Lesson Competency : Create s program that implements the switch statements in accordance with Java framework		
References : Pearson.com. (2019). *Liang, Intro to Java Programming, Comprehensive Version	Pearson.* [online] Available at: https://www.pearson.com/us/higher-education/product/Liang-Intro-to-Java-Programming-Comprehensive-Version-10th-Edition/9780133761313.html [Accessed 18 Jul. 2019].		LAS No.: 15

CONCEPT NOTES

Sample Program using Switch Statement:

A program that prints the corresponding day in a week.

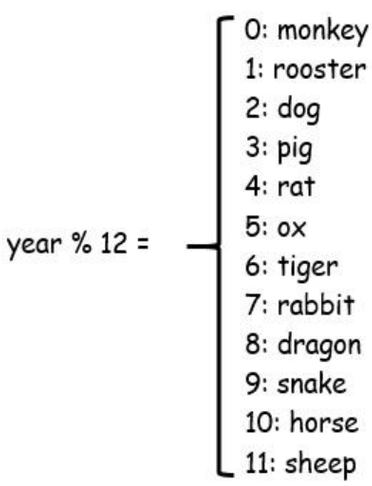
```
public class DaysOfWeek {
public static void main(String[] args) {
{
        int day = 4;
        switch(day) {
        case 1: System.out.println("Monday"); break;
        case 2: System.out.println("Tuesday"); break;
        case 3: System.out.println("Wednesday"); break;
        case 4: System.out.println("Thursday"); break;
        case 5: System.out.println("Friday"); break;
        default: System.out.println("Weekends");
        }
}
```

Output:

Thursday

EXERCISES. Performance Task. Perform the following: (20 points)

Write a program that uses switch statement to find out the Chinese Zodiac sign for a given year. The Chinese Zodiac is based on a twelve-year cycle, with each year represented by an animal— monkey, rooster, dog, pig, rat, ox, tiger, rabbit, dragon, snake, horse, or sheep. Declare a variable named zodiac of type int and set it to any value from 0 -12. Save your program as **ChineseZodiac.java.**



*Note that **year % 12** determines the Zodiac sign. 1900 is the year of the rat because **1900 % 12** is **4**.*

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Looping Statement		
Lesson Competency : Identify the types of looping statements in accordance with Java framework		
References : GeeksforGeeks. (2019). *Loops in Java - GeeksforGeeks*. [online] Available at: https://www.geeksforgeeks.org/loops-in-java/ [Accessed 18 Jul. 2019].		LAS No.: 16

CONCEPT NOTES

Looping Statement can execute a block of code as long as a specified condition is true.

Java provides three ways for executing the loops. While all the ways provide similar basic functionality, they differ in their syntax and condition checking time.

3 Types of Loops in Java

1. While loop - a control flow statement that allows code to be executed repeatedly based on a given Boolean condition.
   Syntax :
   while (boolean condition){
   loop statements;}
2. Do-while loop - is similar to while loop with only difference that it checks for condition after executing the statements, and therefore is an example of **Exit Control Loop**.
   Syntax:
   do
   { statements; }
   while (boolean condition);
3. For loop provides a concise way of writing the loop structure. Unlike a while loop, a for statement consumes the initialization, condition and increment/decrement in one line thereby providing a shorter, easy to debug structure of looping.
   Syntax:
   for(initialization; testing condition; increment/decrement)
   { statements; }

EXERCISES. Identification. Identify the following. Write your answer on the space provided before each number. (2 points each)

_____1. A loop that checks the condition after executing all the statements inside its body.

_____2. It refers to a loop that provides a concise way of writing the loop structure.

_____3. It execute a block of code as long as a specified condition is true.

_____4. A loop that checks the condition at the beginning of the looping structure.

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Looping Statement: While Loop		
Lesson Competency : Create and trace Java programs that implements while loop		
References : GeeksforGeeks. (2019). *Loops in Java - GeeksforGeeks*. [online] Available at: https://www.geeksforgeeks.org/loops-in-java/ [Accessed 18 Jul. 2019].		LAS No.: 17

CONCEPT NOTES

While loop executes statements repeatedly while the condition is true.

Syntax:

```
while (boolean condition) {
        statement(s);
        }
```

Sample Program: A program that prints the even numbers from 1 to 20.

```
public class PrintsEven{
public static void main (String args[]){
        int start = 1;
        while (start <=20)
        {
                if (start % 2==0)
                        System.out.print( start + ", ");
                start++;
        }
}
}
```

Output: 2, 4, 6, 8, 10, 12, 14, 16, 18, 20,

EXERCISES. Performance Task. Perform the following:

1. Write a program that prints the odd numbers from 50 to 100 using while loop. Save your program as **OddNumbersWhileLoop**. (15 points)
2. Identify the output of the following code segment: (5 points)

```
int start = 6;
        while (start <=10)
{
        System.out.println( "Welcome to Java Programming. ");

}
```

Name:	Date:	Score:
Subject : Computer Programming (Java)		
Lesson Title : Looping Statement: Do-While Loop		
Lesson Competency : Create Java programs that implements do- while loop		
References : GeeksforGeeks. (2019). *Loops in Java - GeeksforGeeks*. [online] Available at: https://www.geeksforgeeks.org/loops-in-java/ [Accessed 18 Jul. 2019].		LAS No.: 18

CONCEPT NOTES

Do-While loop is the same as a **while** loop except that it executes the loop body first and then checks the loop boolean condition.

Syntax:

```
do {
        statement(s);
} while (boolean condition);
```

Sample Program: A program that prints the even numbers from 1-20.

```
public class PrintsEven{
public static void main (String args[]){
        int start = 1;

        do{
                if (start % 2==0)
                        System.out.print( start + ", ");
                start++;
        } while (start <=20);

}
}
```

Output: 2, 4, 6, 8, 10, 12, 14, 16, 18, 20,

EXERCISES. Performance Task. Perform the following: (15 points each)

1. Write a program that prints the odd numbers from 20 to 50 using do-while loop. Save your program as **OddNumbersDoWhile.java.**

2. Write a program that prints <Your Name> 20 times using do-while loop. Save your program as **MyNameDoWhile.java.**

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Looping Statement: For Loop		
Lesson Competency : Complete Java programs that implements for loop structure		
References : W3schools.com. (2019). *Java For Loop*. [online] Available at: https://www.w3schools.com/java/java_for_loop.asp [Accessed 18 Jul. 2019].		LAS No.: 19

CONCEPT NOTES

Java For Loop

When you know exactly how many times you want to loop through a block of code, use the for loop instead of a while loop.

Syntax:

```
for(initialization; test condition; increment/decrement)
{
   statements;
}
```

Initialization - is executed (one time) before the execution of the code block.
Test condition - defines the condition for executing the code block.
Increment/decrement - is executed (every time) after the code block has been executed.
Sample Program: A program that prints the numbers from 0 to 4.

```
public class SampleForLoop {
  public static void main(String[] args) {
    for (int i = 0; i < 5; i++) {
      System.out.print(i + ", ");
    }
  }
}
```

Output: 0, 1, 2, 3, 4,

EXERCISES.

1. Complete the following code segment such that it will print "Hello" 5 times using for loop. (2 points each)

```
[      ] (int start = 0; start < 5; [      ]) {
    System.out.println([      ]);
  }
```

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Java Array		
Lesson Competency : Create and use Arrays in accordance with Java framework		
References : W3schools.com. (2019). *Java For Loop.* [online] Available at: https://www.w3schools.com/java/java_arrays.asp [Accessed 18 Jul. 2019].		LAS No.: 20

CONCEPT NOTES

Java Arrays are collection (sequence) of a fixed number of variables called elements or components, wherein all the elements are of the same data type. A one-dimensional array is an array in which the elements are arranged in a list form.

Syntax for declaring an array:

dataType[] arrayName = new dataType[intExp];

The statement:

int[] num = new int[5];

- declares and creates the array num consisting of 5 elements (5 is the size of the array). Each element is of type int. The elements are accessed as num[0], num[1], num[2], num[3], and num[4].

To set the value of the array:

num[0] = 5; - sets the first value of the array to 5

num[4] = 8; - sets the last value of the array to 8, the rest has a default value of 0.

You can also declare an array with default values:

double[] num = {1.1, 3.4, 2.5, 5.6};

String[] cars = {"Toyota", "Isuzu", "Mitsubishi"};

EXERCISES. Answer the following questions. (3 points each)

1. Create an array of type double named grades of size 5.
2. Set the 2nd value of array grades to 98.
3. Set the last value of array grades to 87.
4. Declare an array of type String named country that contains the values, "Spain", "Japan", and "Korea".
5. Set the 3rd value of array country to "Taiwan".

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Java Array: Displaying elements		
Lesson Competency : Create and use Arrays in accordance with Java framework		
References : W3schools.com. (2019). *Java For Loop.* [online] Available at: https://www.w3schools.com/java/java_arrays.asp [Accessed 18 Jul. 2019].		LAS No.: 21

CONCEPT NOTES

Given an array named num of type int:

```
int[] num = {5, 8 ,2, 3, 6};
```

Displaying Elements: To print an array, you have to print each element in the array using a loop like the following:

```
for (int i = 0; i < num.length; i++) {
    System.out.print(num[i] + " ");
}
```

Output: 5 8 2 3 6

Summing up array elements:

```
int sum = 0;
for (int i = 0; i < num.length; i++) {
    sum = sum + num [i];
    System.out.print(sum);
}
```

Output: 24

EXERCISES. Perform the following.

1. Write a complete program that performs the following. Save your program as **MyFirstArray.java**. (20 points)
   a. Declare an array named myScore of type int and sets its value to 15, 22, 18, 20 and 23.
   b. Print the elements stored in your array.
   c. Calculate and print the average of the scores stored in your array.
2. Consider the following declaration:
   double[] salary = new double[10];
   In this declaration, identify the following: (2 points each)
   a. Array name.
   b. Array size.
   c. Data type of each array component.
   d. Range of values for the index of the array.

Name:	Date:	Score:
Subject : Computer Programming (Java)		
Lesson Title : Basic Object Oriented Concept: Inheritance		
Lesson Competency : Identify concepts of inheritance in accordance with Java framework		
References : www.javatpoint.com. (2019). *Inheritance in Java - Javatpoint*. [online] Available at: https://www.javatpoint.com/inheritance-in-java [Accessed 19 Jul. 2019].		LAS No.: 22

CONCEPT NOTES

Inheritance in Java is a mechanism in which one object acquires all the properties and behaviors of a parent object.

- you can create new classes that are built upon existing classes.
- reuse methods and fields of the parent class.
- can add new methods and fields in your current class.
- represents the **IS-A relationship** which is also known as a *parent-child* relationship.

Why use inheritance in java

- For Method Overriding (so runtime polymorphism can be achieved).
- For Code Reusability.

Terms used in Inheritance

- **Class:** a group of objects which have common properties. It is a template or blueprint from which objects are created.
- **Sub Class/Child Class:** a class which inherits the other class. It is also called a derived class, extended class, or child class.
- **Super Class/Parent Class:** the class from where a subclass inherits the features. It is also called a base class or a parent class.
- **Reusability:** is a mechanism which facilitates you to reuse the fields and methods of the existing class when you create a new class. You can use the same fields and methods already defined in the previous class.

EXERCISES. Identification. Identify the following. Write your answer on the space provided before each number. (2 points each)

_____1. It is also known as parent-child relationship.

_____2. A class from where a subclass inherits the features.

_____3. A mechanism in which one object acquires all the properties and behaviors of a parent object.

_____4. It is a template or blueprint from which an object is created.

_____5. A class which inherits the other class.

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC IIIS		
Lesson Title : Basic Object Oriented Concept: Inheritance		
Lesson Competency : Implement inheritance in accordance with Java framework		
References : www.javatpoint.com. (2019). *Inheritance in Java - Javatpoint*. [online] Available at: https://www.javatpoint.com/inheritance-in-java [Accessed 19 Jul. 2019].		LAS No.: 23

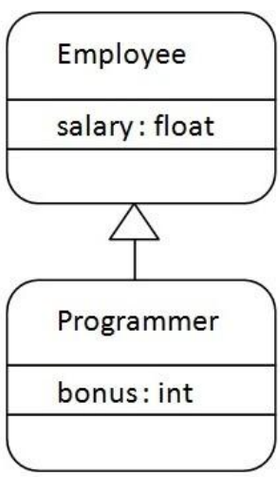
CONCEPT NOTES

Syntax of Java Inheritance

```
class Subclass_name extends Superclass_name
    {
    //methods and fields
    }
```

extends keyword - indicates that you are making a new class that derives from an existing class. The meaning of "extends" is to increase the functionality.

Java Inheritance Example



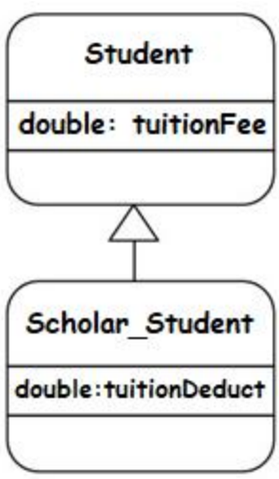

```
class Employee{
    float salary=40000;
}
class Programmer extends Employee{
     int bonus=10000;
 public static void main(String args[]){
     Programmer p=new Programmer();
     System.out.println("Programmer salary is:"+p.salary);
     System.out.println("Bonus of Programmer is:"+p.bonus);
}
}
```

Output:

Programmer salary is:40000.0
Bonus of programmer is:10000

EXERCISES. Performance task. Perform the following. (30 points)

Given the diagram on the right side, create a program that uses inheritance. Set the tuitionFee to 26,500 in your super class and set tuitionDeduct to 50% of the tuitionFee in your subclass. Provide a formula in order to compute the balance of the scholar student. Display tution fee, amount deducted to the tuition fee and the computed balance. Save your program as **Programmer.java**.



Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Basic object oriented concept: Encapsulation		
Lesson Competency : Write programs that uses encapsulation in accordance with Java framework		
References : W3schools.com. (2019). *Java Encapsulation and Getters and Setters.* [online] Available at: https://www.w3schools.com/java/java_encapsulation.asp [Accessed 19 Jul. 2019].		LAS No.: 24

CONCEPT NOTES

Encapsulation means to make sure that "sensitive" data is hidden from users. To achieve this, you must:

- declare class variables/attributes as private (only accessible within the same class)
- provide public **setter** and **getter** methods to access and update the value of a private variable

Private variables can only be accessed within the same class (an outside class has no access to it). However, it is possible to access them if we provide public **getter** and **setter** methods.

get method - returns the variable value,

set method - sets the value.

Syntax for both is that they start with either get or set, followed by the name of the variable, with the first letter in upper case:

Sample Program:

```
public class Person {
  private String name;        //private = restricted access

  public String getName() {   // Getter
    return name;
  }

  public void setName(String newName) {   // Setter
    this.name = newName;
  }
}
```

EXERCISES. Using the program above, re-write the code and do the following: (15 points)

a. Declare another private variable named age of type int.

b. Provide a get and set method for the variable age.

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Handling Exceptions		
Lesson Competency : Determine the functions of Java exceptions parts		
References : W3schools.com. (2019). *Java Exceptions (Try...Catch)*. [online] Available at: https://www.w3schools.com/java/java_try_catch.asp [Accessed 19 Jul. 2019].		LAS No.: 25

CONCEPT NOTES

Java Exceptions

When executing Java code, different errors can occur: coding errors made by the programmer, **errors due to wrong input,** or other unforeseeable things. When an error occurs, Java will normally stop and generate an error message. The technical term for this is: Java will throw an **exception** (throw an error).

Java try and catch

try statement - allows you to define a block of code to be tested for errors while it is being executed.
catch statement - allows you to define a block of code to be executed, if an error occurs in the try block.

The try and catch keywords come in pairs:

Syntax:

```
try {
  //  Block of code to try
}
catch(Exception e) {
  //  Block of code to handle errors
}
```

EXERCISES

Mark the following as True or False. Write your answer on the space provided before each number. (2 points each)

_____1. Java never throws an error because it has an exception.
_____2. The catch statement defines a block of code to be tested for errors.
_____3. The try statement defines a code to be executed.
_____4. Errors are due to wrong input and other unforeseeable things.
_____5. Try and catch keywords come in pairs.

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Handling Exceptions: Sample Program		
Lesson Competency : Determine how exceptions handle and alter normal program flow		
References : W3schools.com. (2019). *Java Exceptions (Try...Catch)*. [online] Available at: https://www.w3schools.com/java/java_try_catch.asp [Accessed 19 Jul. 2019].		LAS No.: 26

CONCEPT NOTES

Consider the following example:
This will generate an error, because **myNumbers[10]** does not exist.

```
public class MyClass {
  public static void main(String[ ] args) {
    int[] myNumbers = {1, 2, 3};
    System.out.println(myNumbers[10]); // error!
  }
}
```

The output will be something like this:

Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 10
at MyClass.main(MyClass.java:4)

To handle errors, the following program uses try and catch.

Example:

```
public class MyClass {
  public static void main(String[ ] args) {
    try {
      int[] myNumbers = {1, 2, 3};
      System.out.println(myNumbers[10]);
    } catch (Exception e) {
      System.out.println("Something went wrong.");
    }
  }
}
```

The output will be: **Something went wrong.**

EXERCISES. Discussion. Discuss the following: (5 points each)

1. How to handle unexpected errors in Java programming?

2. How do exceptions alter normal program flow?

DepED DEPARTMENT OF EDUCATION K TO 12 PROGRAM

SHARED OPTIONS

SENIOR HIGH ALTERNATIVE RESPONSIVE EDUCATION DELIVERY

GRADE 11 DLP LEARNING ACTIVITY SHEET

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Object-Oriented Software Development (OOSD)		
Lesson Competency : Identify the concepts of Object-Oriented Software Development (OOSD) process in accordance with Java framework		
References : rmuniv.ac.in. (2019). [online] Available at: http://www.srmuniv.ac.in/sites/default/files/files/system_development.pdf [Accessed 19 Jul. 2019].		LAS No.: 27

CONCEPT NOTES

Object Oriented System Development (OOSD)

- a way to develop software by building self – contained modules or objects that can be easily replaced, modified and reused.
- Software - a collection of discrete objects that encapsulate their data as well as the functionality of model real-world events "objects" and emphasizes its cooperative philosophy by allocating tasks among the objects of the applications.
- Class - an object oriented system carefully delineates between its interface and the implementation of that interface (how the class does what it does).

Need of Object Orientation

- ***Allows higher level of abstraction:*** Makes designing, coding, testing & maintaining the system much simpler.
- ***Provides Seamless transition among different phases of software development:*** This reduces the level of complexity and redundancy and makes for clearer, more robust system development.
- ***Encourage good development techniques:*** In a properly designed system, the routines and attributes within a class are held together tightly, the classes will be grouped into subsystems but remain independently and therefore, changing one class has no impact on other classes and so, the impact is minimized.
- ***Promotes of reusability:*** Objects are reusable because they are modeled directly out of a real – world problem domain.

EXERCISES. Identification. Identify the following. Write your answer on the space provided before each number. (2 points each)

_____1. It refers to an object oriented system carefully delineates between its interface.

_____2. A collection of discrete objects.

_____3. It is a way to develop software by building self – contained modules.

_____4. It refers to how class does what it does.

_____5. It makes designing, coding, testing & maintaining the system much simpler.

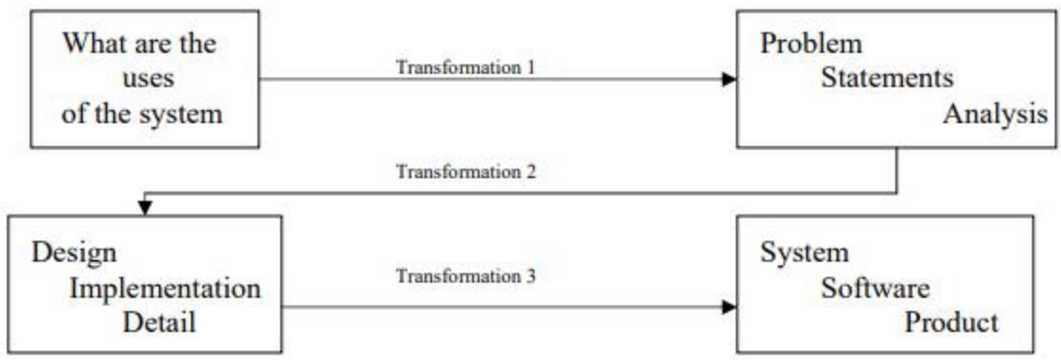
Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : OOSD Life Cycle		
Lesson Competency : Explain Object-Oriented Software Development (OOSD) process in accordance with Java framework		
References : Srmuniv.ac.in. (2019). [online] Available at: http://www.srmuniv.ac.in/sites/default/files/files/system_development.pdf [Accessed 19 Jul. 2019].		LAS No.: 28

CONCEPT NOTES

Object – Oriented System Development Life Cycle

-consists of **analysis, design, implementation, testing and refinement**. Its main aim is **to transform users' needs into a software solution.**

The development is a process of change, refinement, transformation or addition to the existing product. The software development process can be viewed as a series of transformations, where the output of one transformation becomes input of the subsequent transformation.

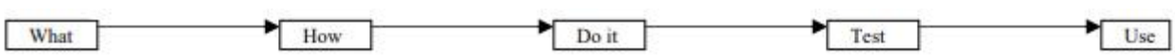


Transformation 1 (analysis) translates the users' needs into system requirements & responsibilities.

Transformation 2 (design) begins with a problem statement and ends with a detailed design that can be transformed into an operational system. It includes the bulk of s/w development activity.

Transformation 3 (implementation) refines the detailed design into the system deployment that will satisfy the users' needs. It represents embedding s/w product within its operational environment. o

An example of s/w development process is the waterfall approach which can be stated as below



EXERCISES. Discussion. Discuss the following: (10 points)

1. Explain the OOSD Life Cycle in your own words.

DepED DEPARTMENT OF EDUCATION K TO 12 PROGRAM

SHARED OPTIONS
SENIOR HIGH ALTERNATIVE RESPONSIVE EDUCATION DELIVERY
GRADE 11 DLP LEARNING ACTIVITY SHEET

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Benefits of Modeling Software		
Lesson Competency : Explain benefits of modeling software in accordance with Java framework		
References : Cabot, J. (2019). *List of the (supposed) benefits of software modeling.* [online] Modeling Languages. Available at: https://modeling-languages.com/list-supposed-benefits-software-modeling/ [Accessed 19 Jul. 2019].		LAS No.: 29

CONCEPT NOTES

Systems modeling or **software modeling** is the interdisciplinary study of the use of models to conceptualize and construct systems in business and IT development.

Benefits of Software Modeling

- It improves the productivity of the development team
- It reduces the number of defects in the final code
- It improves the understandability of the system (which eases the integration of new team members)
- It increases the decomposition and modularization of the system
- It facilitates the system's evolution AND maintenance
- If facilitates the reuse OF parts OF the system IN new projects

EXERCISES

I. Mark the following as True or False. Write your answer on the space provided before each number. (2 points each)

_____1. Software modeling uses pictures to conceptualize and constructs systems.

_____2. Software modeling improves the efficiency of the development team.

_____3. Software modeling decreases decomposition of the system.

_____4. Software modeling facilitates systems evolution and maintenance.

_____5. Software modeling reduces the number of defects in the final code.

II. Discussion. Discuss the following: (10 points)

1. Select one benefit of Software modeling and explain in your own words.

DepED DEPARTMENT OF EDUCATION K TO 12 PROGRAM

SHARED OPTIONS

SENIOR HIGH ALTERNATIVE RESPONSIVE EDUCATION DELIVERY

GRADE 11 DLP LEARNING ACTIVITY SHEET

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Use Case Diagram		
Lesson Competency: Define Use Case Diagram and justify the need for a Use Case Diagram in accordance with Java framework		
References : https://www.smartdraw.com/use-case-diagram/		LAS No.: 30

CONCEPT NOTES

Use case diagram is dynamic or behavior diagram in Unified Modeling Language (UML). It models the functionality of a system using actors and use cases.

- set of actions, services, and functions that the system needs to perform. In this context, a "system" is something being developed or operated, such as a web site. The "actors" are people or entities operating under defined roles within the system.

- diagrams are valuable for visualizing the functional requirements of a system that will translate into design choices and development priorities.
- help identify any internal or external factors that may influence the system and should be taken into consideration.
- provide a good high level analysis from outside the system. Use case diagrams specify how the system interacts with actors without worrying about the details of how that functionality is implemented.

EXERCISES

Answer the following questions: (5 points each)

1. Define a use case diagram.

2. Why use case diagrams are needed before developing a system?

DepED DEPARTMENT OF EDUCATION K TO 12 PROGRAM

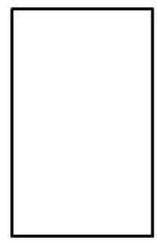
SHARED OPTIONS

SENIOR HIGH ALTERNATIVE RESPONSIVE EDUCATION DELIVERY

GRADE 11 DLP LEARNING ACTIVITY SHEET

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Use Case Diagram		
Lesson Competency: Identify the different symbols used in creating a use case diagram		
References : https://www.smartdraw.com/use-case-diagram/#whatisUseCase https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-use-case-diagram/		LAS No.: 31

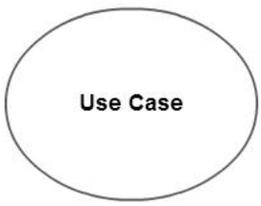
CONCEPT NOTES



Symbols use in Use CASE Diagram

System

Draw your system's boundaries using a rectangle that contains use cases. Place actors outside the system's boundaries.



Use Case

Draw use cases using ovals. Label the ovals with verbs that represent the system's functions.

Actors

Actors are the users of a system. When one system is the actor of another system, label the actor system with the actor stereotype.

Communication Link

The participation of an actor in a use case is shown by connecting an actor to a use case by a solid link. Actors may be connected to use cases by associations, indicating that the actor and the use case communicate with one another using messages.

EXERCISES

Answer the following question: (10 pts)

1. What are the symbols used in creating a use case diagram? Draw and label your answer.

DepED DEPARTMENT OF EDUCATION K TO 12 PROGRAM

SHARED OPTIONS

SENIOR HIGH ALTERNATIVE RESPONSIVE EDUCATION DELIVERY

GRADE 11 DLP LEARNING ACTIVITY SHEET

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Developing CASE diagram for a software system		
Lesson Competency: Develop use case diagram		
References : https://www.lucidchart.com/pages/uml-use-case-diagram https://online.visual-paradigm.com/diagrams/tutorials/use-case-diagram-tutorial/		LAS No.: 32

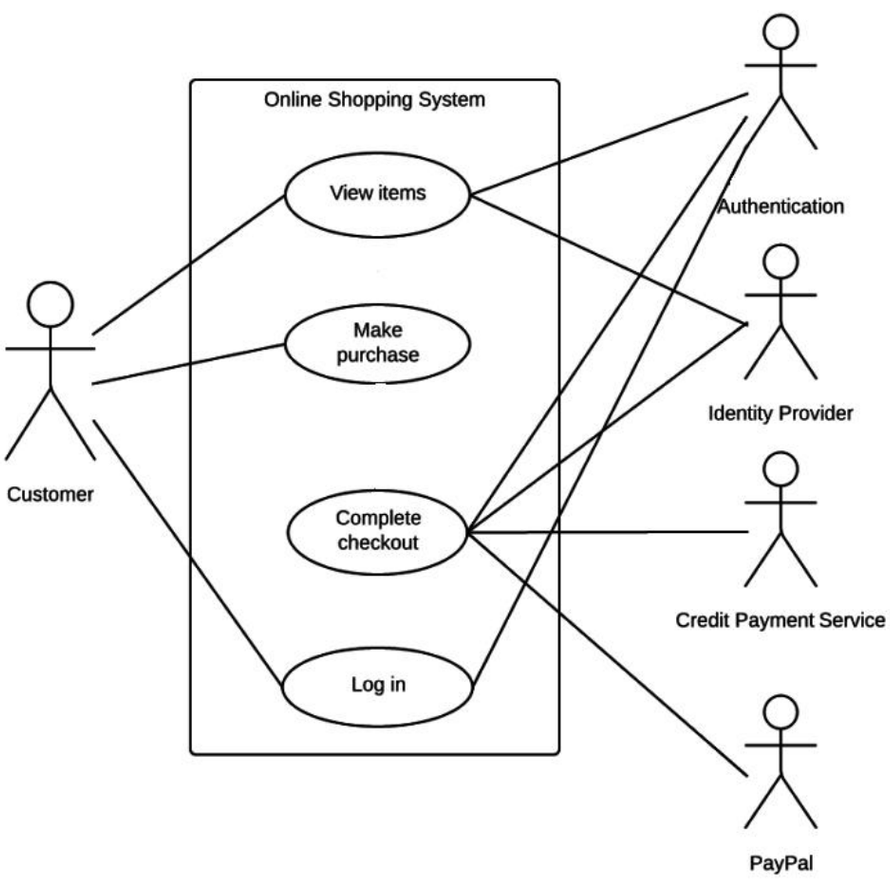
CONCEPT NOTES

How to Draw a Use Case Diagram?

A Use Case model can be developed by following the steps below.

1. Identify the Actors (role of users) of the system.
2. For each category of users, identify all roles played by the users relevant to the system.
3. Identify what are the users required the system to be performed to achieve these goals.
4. Create use cases for every goal.
5. Structure the use cases.
6. Prioritize, review, estimate and validate the users.

Online shopping use case diagram example



EXERCISES.

Vending Machine

After a client interview the following system scenarios were identified:

- A customer buys a product
- The supplier restocks the machine
- The supplier collects money from the machine

Answer the following: (20 points)

1. Identify the actor(s) of the above scenario.
2. Create a use case diagram for the above scenario.

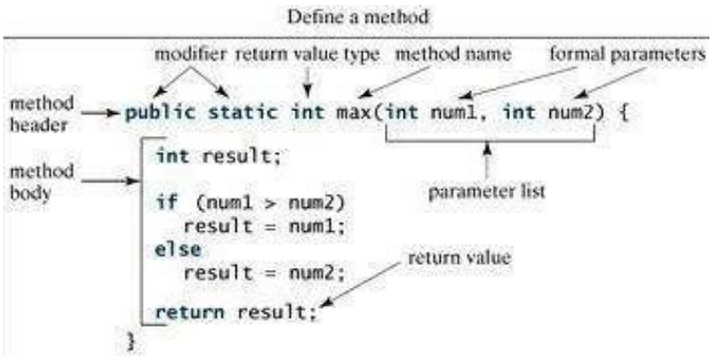
Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Java Methods		
Lesson Competency : Identify the parts of a method and write a method header in accordance with Java framework		
References : W3schools.com. (2019). *Java Methods*. [online] Available at: https://www.w3schools.com/java/java_methods.asp [Accessed 19 Jul. 2019].		LAS No.: 33

CONCEPT NOTES

Java Methods - A is a block of code which only runs when it is called. You can pass data, known as parameters, into a method. Methods are used to perform certain actions, and they are also known as **functions**.

Why use methods? To reuse code: define the code once, and use it many times.

Parts of Method



Creating a Method

A method must be declared within a class. It is defined with the name of the method, followed by parentheses **()**. Java provides some pre-defined methods, such as System.out.println(), but you can also create your own methods to perform certain actions:

Example: Create a method named myMethod() inside MyClass:

```
public class MyClass {
  static void myMethod() {
    // code to be executed
  }
}
```

- myMethod()- name of the method
- static means that the method belongs to the MyClass class and not an object of the MyClass class. You will learn more about objects and how to access methods through objects later in this tutorial.
- void means that this method does not have a return value.

EXERCISES. Answer the following:

1. Given:

   ```
   public static void printAverage(){
   }
   ```

   a. What is the name of the method? (2 points) _____
   b. What is the data type of the method? (2 points) _____
2. Write a method header named **computeAverage()** of type double that holds two parameters of type double. (5 points)

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Java methods with arguments and return values		
Lesson Competency : Create methods with arguments and return values in accordance with Java framework		
References : W3schools.com. (2019). *Java Methods*. [online] Available at: https://www.w3schools.com/java/java_methods.asp [Accessed 19 Jul. 2019].		LAS No.: 34

CONCEPT NOTES

Return Values

If you want the method to return a value, you can use a primitive data type (such as `int`, `char`, double etc.) instead of void, and use the return keyword inside the method:

Example:

```
public class MyClass {
  static int myMethod(int x) {
    return 5 + x;
  }
  public static void main(String[] args) {
    System.out.println(myMethod(3));
  }
}
```

Method header

Method definition or body – what method will do

Output: 8

The following example returns the sum of a method's **two parameters**:

Example:

```
public class MyClass {
  static int myMethod(int x, int y) {
    return x + y;
  }
  public static void main(String[] args) {
    System.out.println(myMethod(5, 3));
  }
}
```

Output: 8

EXERCISES. Perform the following: (15 points)

1. Write the definition of a method named **computeResult()** that takes three numbers of type int and returns the sum of the first two numbers multiplied by the third number.

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Methods with arguments and return values		
Lesson Competency : Create methods with arguments and return values in accordance with Java framework		
References : W3schools.com. (2019). *Java Methods*. [online] Available at: https://www.w3schools.com/java/java_methods.asp [Accessed 19 Jul. 2019].		LAS No.: 35

CONCEPT NOTES

A Method with If...Else

It is common to use if-else statements inside methods:

```
public class MyClass {
        // Create a checkAge() method with an integer variable called age
  static void checkAge(int age) {
        // If age is less than 18, print "access denied"
    if (age < 18) {
      System.out.println("Access denied - You are not old enough!");
        // If age is greater than 18, print "access granted"
    } else {
      System.out.println("Access granted - You are old enough!");
    }
  }
  public static void main(String[] args) {
    checkAge(20);   // Call the checkAge method and pass along an age of 20
  }
}
```

Output: Access granted - You are old enough

EXERCISES. Do the following, save your program as **ValidateAgeLevel.java**. Follow the comments to insert the missing parts of the code below: (20 points)

```
public class ValidateAgeLevel()
        // Create a checkAge() method with an integer variable called age
        _____{
        // If age is less than or equal to 11, print "You are an Elementary pupil."
        _____
        _____
        //If age is less than or equal to 16, print "You are a Junior High Student."
        _____
        _____
        //If age is above 16, print "You are a Senior High or College Student."
        _____
        _____
        }
}
public static void main(String[] args) {
        // Call the checkAge method and pass along an age of 20
         _____}}
```

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Static keywords and its applications		
Lesson Competency : Identify the concepts in using static keywords to methods and fields in accordance with Java framework		
References : GeeksforGeeks. (2019). *static keyword in java - GeeksforGeeks*. [online] Available at: https://www.geeksforgeeks.org/static-keyword-java/ [Accessed 19 Jul. 2019].		LAS No.: 36

CONCEPT NOTES

Static keyword in Java is a non-access modifier.

To create a static member (block,variable,method,nested class), precede its declaration with the keyword *static*. When a member is declared static, it can be accessed before any objects of its class are created, and without reference to any object.

For example, in below java program, we are accessing static method m1() without creating any object of ***Test*** class.

```
// Java program to demonstrate that a static member
// can be accessed before instantiating a class
class Test
{
   // static method
   static void m1()
   {
      System.out.println("from m1");
   }
    public static void main(String[] args)
   {
       // calling m1 without creating any object of class Test
      m1();
   }
}
```

Output: from m1

EXERCISES. Mark the following as True or False. Write your answer on the space provided before each number. (2 points each)

_____1. In using static keyword, creating object of a class is not needed.
_____2. Static keyword is an access modifier in Java.
_____3. Static keyword is applicable only in blocks.
_____4. When a member is declared static, it can be accessed before any objects of its class are created.
_____5. Using static keyword, saves memory.

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Static keywords and its applications: Static Variables		
Lesson Competency : Identify output of programs that uses static keywords to methods and fields in accordance with Java framework		
References : GeeksforGeeks. (2019). *static keyword in java - GeeksforGeeks*. [online] Available at: https://www.geeksforgeeks.org/static-keyword-java/ [Accessed 19 Jul. 2019].		LAS No.: 37

CONCEPT NOTES

Static variables

When a variable is declared as static, then a single copy of variable is created and shared among all objects at class level. They are global variables.

Important points for static variables

- We can create static variables at class-level only.
- static block and static variables are executed in order they are present in a program.

Sample Program

```
class Test
{
   static int a = m1();   // static variable
   static {               // static block
      System.out.println("Inside static block");
   }
   static int m1() {      // static method
      System.out.println("from m1");
      return 20;
   }
   public static void main(String[] args)      // static method(main !!)
   {
     System.out.println("Value of a : "+a);
     System.out.println("from main");
   }
 }
```

Output:

```
from m1
Inside static block
Value of a : 20
from main
```

EXERCISES. Tracing Output. Identify the output of the following program without using a Java Compiler. (15 points)

```
class Programmer
{
   static int num = myName();
   static int myName() {
      System.out.println("Jones");
      return 15;
   }
   static {
      System.out.println("Welcome");
   }
   public static void main(String[] args)
   {
     System.out.println("Java Programming");
     System.out.println("Value of num : "+num);

   }
 }
```

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Method Overloading		
Lesson Competency : Create overloaded method in accordance with Java framework		
References : https://www.javatpoint.com/method-overloading-in-java		LAS No.: 38

CONCEPT NOTES

Method Overloading is a class containing multiple methods having same name but different in parameters.

It *increases the readability of the program.*

Ways to overload the method in java

1. By changing number of arguments
2. By changing the data type

Method Overloading: changing number of arguments

Sample Program:

```
class Adder{
    static int add(int a,int b){
        return a+b;
    }
    static int add(int a,int b,int c){
        return a+b+c;
    }
}
class TestOverloading1{
public static void main(String[] args){
    System.out.println(Adder.add(11,11));
    System.out.println(Adder.add(11,11,11));
    }
}
```

EXERCISES.

Using the sample program above, perform the following.

1. Declare a new method named computeAverage() that holds two parameters of type int named num1 and num2. Provide a formula in order to compute the average and return the result. (10 points)
2. Overload the method computeAverage() by adding new parameter of type int named num3. Provide a formula in order to compute the average and return the result. (10 points)
3. Call your methods inside main(). (5 points)

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Method Overloading: Changing Data Type of Arguments		
Lesson Competency : Create overloaded method in accordance with Java framework		
References : https://www.javatpoint.com/method-overloading-in-java		LAS No.: 39

CONCEPT NOTES

Method Overloading: Changing Data Type of Arguments

In this example, we have created two methods that differs in data type. The first add method receives two integer arguments and second add method receives two double arguments.

```
class Adder{
    static int add(int a, int b){
        return a+b;
    }
    static double add(double a, double b){
        return a+b;
    }
}
class TestOverloading2{
public static void main(String[] args){
    System.out.println(Adder.add(11,11));
    System.out.println(Adder.add(12.3,12.6));
    }
}
```

EXERCISES

Using the sample program above, perform the following.

1. Declare a new method named **computeTotal()** that holds two parameters of type int named **n1** and **n2**. Provide a formula in order to compute the total and return the result. (10 points)
2. Overload the method **computeTotal()** by changing the data type of the two parameters into double. Provide a formula in order to compute the total and return the result. (10 points)
3. Call your methods inside main(). (5 points)

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Read and Write Data from the Console		
Lesson Competency : Identify the different method used by Scanner class.		
References : https://www.w3schools.com/java/java_user_input.asp		LAS No.: 40

CONCEPT NOTES

Java User Input (Scanner) – The **Scanner** class is used to get user input, and it is found in the java.util package.

Input Types

Method	**Description**
nextBoolean()	Reads a boolean value from the user
nextByte()	Reads a byte value from the user
nextDouble()	Reads a double value from the user
nextFloat()	Reads a float value from the user
nextInt()	Reads a int value from the user
nextLine()	Reads a String value from the user
nextLong()	Reads a long value from the user
nextShort()	Reads a short value from the user

EXERCISES. Identify the following. Write your answer on the space provided before each number. (1 pt each)

_____1. It is a class that is used to get input from the user.
_____2. What method is used to read a short value from the user?
_____3. What method is used to read a long value from the user?
_____4. What method is used to read a String value from the user?
_____5. What method is used to read a byte value from the user?
_____6. What method is used to read a boolean value from the user?
_____7. What value is returned if we use the method nextLine?
_____8. What value is returned if we use the method nextByte?
_____9. What value is returned if we use the method nextDouble?
_____10. What value is returned if we use the method nextInt?

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Read and Write Data from the Console		
Lesson Competency : Write programs that accepts input from the user		
References : https://www.w3schools.com/java/java_user_input.asp		LAS No.: 41

CONCEPT NOTES

Sample Program that uses Scanner

In the example below, it uses different methods to read data of various types:

```
import java.util.Scanner;

class MyClass {
  public static void main(String[] args) {
    Scanner console = new Scanner(System.in);

    System.out.println("Enter name, age and salary");

    String name = console.nextLine();          // String input
    int age = console.nextInt();               // Numerical input
    double salary = console.nextDouble();      // Numerical input

    // Output input by user
    System.out.println("Name: " + name);
    System.out.println("Age: " + age);
    System.out.println("Salary: " + salary);
  }
}
```

EXERCISES. Performance Task. Perform the following.

1. Write a program that prompts the user to input a number. The program should then output the number and a message saying whether the number is positive, negative, or zero. Save your program as **TryMyNumber.java**. (20 points)
2. Revise your program such that the user will be given a chance to re-enter a number about 5 times at the same time displaying the message if the number entered is positive, negative or zero. Save your program as **TryMyNumberLoop.java**. (20 points)

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Java Files		
Lesson Competency : Identify the methods for creating and getting information about files.		
References : https://www.w3schools.com/java/java_files.asp		LAS No.: 42

CONCEPT NOTES

The File class from the java.io package, allows us to work with files.
To use the File class, create an object of the class, and specify the filename or directory name:

Example:

```
import java.io.File;  // Import the File class
File myObj = new File("filename.txt"); // Specify the filename
```

Method	Return Type	Description
canRead()	Boolean	Tests whether the file is readable or not
canRead()	Boolean	Tests whether the file is writable or not
createNewFile()	Boolean	Creates an empty file
delete()	Boolean	Deletes a file
exists()	Boolean	Tests whether the file exists
getName()	String	Returns the name of the file
getAbsolutePath()	String	Returns the absolute pathname of the file
length()	Long	Returns the size of the file in bytes
list()	String[]	Returns an array of the files in the directory
mkdir()	Boolean	Creates a directory

EXERCISES

Identify the following. Write your answer on the space provided before each number. (2 points each)

_____1. What method is use to return the name of the file?
_____2. What is the return type of list()?
_____3. What method is used to create and empty file?
_____4. What is the return type of getName() method?
_____5. In what package does File in Java is located?

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Java Files: Create a File		
Lesson Competency : Write a program that create a file.		
References : https://www.w3schools.com/java/java_files.asp		LAS No.: 43

CONCEPT NOTES

Create a File

Use the createNewFile() method to create a file. This method returns a boolean value: true if the file was successfully created, and false if the file already exists. Note that the method is enclosed in a try...catch block. This is necessary because it throws an IOException if an error occurs (if the file cannot be created for some reason:

Sample Program:

```
import java.io.File;              // Import the File class
import java.io.IOException;  // Import the IOException class to handle errors
public class CreateFile {
  public static void main(String[] args) {
    try {
      File myObj = new File("filename.txt");
      if (myObj.createNewFile()) {
        System.out.println("File created: " + myObj.getName());
      } else {
        System.out.println("File already exists.");
      }
    } catch (IOException e) {
      System.out.println("An error occurred.");
      e.printStackTrace();
    }
  }
}
```

EXERCISES. Perform the following. (20 points)

1. Write a program that creates a file named myFirstTextFile.txt. If the file is successfully created, display a message "Successfully created a file". Otherwise, display a message "File already exists." In case an error is encountered, display a message "An error exists". Save your program as **MyCreateFileProgram.java**.

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Java Files: Write to a File		
Lesson Competency: Create a program that writes to an existing file.		
References : https://www.w3schools.com/java/java_files.asp		LAS No.: 44

CONCEPT NOTES

Write To a File

In the following example, we use the FileWriter class together with its write() method to write some text to the file we created in the example above. Note that when you are done writing to the file, you should close it with the close() method:

Sample Program:

```
import java.io.FileWriter;   // Import the FileWriter class
import java.io.IOException;  // Import the IOException class to handle errors

public class WriteToFile {
  public static void main(String[] args) {
    try {
      FileWriter myWriter = new FileWriter("filename.txt");
      myWriter.write("Files in Java might be tricky, but it is fun enough!");
      myWriter.close();
      System.out.println("Successfully wrote to the file.");
    } catch (IOException e) {
      System.out.println("An error occurred.");
      e.printStackTrace();
    }
  }
}
```

EXERCISES. Perform the following. (20 points)

1. Create a program that writes the following sentences to the file you created in the previous activity:

   Java is an interesting programming language.
   It is easy to learn.

   If writing of lines is successful, display a message "You had successfully wrote to the file." If an error occurred, display a message "You encountered an error". Save your program as **MyWriteToFileProgram.java**.

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Java Files: Read a File		
Lesson Competency: Create a program that reads the content of a certain file.		
References : https://www.w3schools.com/java/java_files.asp		LAS No.: 45

CONCEPT NOTES

Read a File

In the following example, we use the Scanner class to read the contents of the text file we created in the example above:

```
import java.io.File;  // Import the File class
import java.io.FileNotFoundException;  // Import this class to handle errors
import java.util.Scanner; // Import the Scanner class to read text files

public class ReadFile {
  public static void main(String[] args) {
    try {
      File myObj = new File("filename.txt");
      Scanner myReader = new Scanner(myObj);
      while (myReader.hasNextLine()) {
        String data = myReader.nextLine();
        System.out.println(data);
      }
      myReader.close();
    } catch (FileNotFoundException e) {
      System.out.println("An error occurred.");
      e.printStackTrace();
    }
  }
}
```

EXERCISES. Perform the following. (20 points)

1. Create a program that reads and displays the content of the file you used in the previous activity. If an error occurred, display a message "You encountered an error". Save your program as **MyReadFileProgram.java**.

DepED DEPARTMENT OF EDUCATION K TO 12 PROGRAM

SHARED OPTIONS

SENIOR HIGH ALTERNATIVE RESPONSIVE EDUCATION DELIVERY

GRADE 11 DLP LEARNING ACTIVITY SHEET

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Java Access Modifier: Default Access Modifier		
Lesson Competency: Identify the different access modifiers in Java		
References : https://www.tutorialspoint.com/java/java_access_modifiers		LAS No.: 46

CONCEPT NOTES

Java provides a number of access modifiers to set access levels for classes, variables, methods, and constructors. The four access levels are –

- Visible to the package, the default. No modifiers are needed.
- Visible to the class only (private).
- Visible to the world (public).
- Visible to the package and all subclasses (protected).

Default Access Modifier - No Keyword

Default access modifier - means we do not explicitly declare an access modifier for a class, field, method, etc.

A variable or method declared without any access control modifier is available to any other class in the same package. The fields in an interface are implicitly public static final and the methods in an interface are by default public.

Example

Variables and methods can be declared without any modifiers, as in the following examples –

```
String version = "1.5.1";
boolean processOrder() {
   return true;
}
```

EXERCISES. Identify the following. Write your answer on the space provided before each number. (2 points each)

_____1. What keyword is used to make classes, variables, methods, and constructors visible to the package only?

_____2. What keyword is used to make classes, variables, methods, and constructors visible to the world?

_____3. What keyword is used to make classes, variables, methods, and constructors visible to the class only?

_____4. What keyword is used to make classes, variables, methods, and constructors visible to the package and subclasses?

_____5. How many access modifiers are there in Java?

DepED DEPARTMENT OF EDUCATION K TO 12 PROGRAM

SHARED OPTIONS

SENIOR HIGH ALTERNATIVE RESPONSIVE EDUCATION DELIVERY

GRADE 11 DLP LEARNING ACTIVITY SHEET

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Java Access Modifier: Private Access Modifier		
Lesson Competency: Identify the concepts applicable in using private access modifier.		
References : https://beginnersbook.com/2013/05/java-access-modifiers/		LAS No.: 47

CONCEPT NOTES

Private access modifier

The scope of private modifier is limited to the class only.

1. Private Data members and methods are only accessible within the class
2. Class and Interface cannot be declared as private
3. If a class has private constructor then you cannot create the object of that class from outside of the class.

Sample Program

The example below throws compilation error because we are trying to access the private data member and method of class ABC in the class Example. The private data member and method are only accessible within the class.

```
class ABC{
   private double num = 100;
   private int square(int a){
        return a*a;
   }
}
public class Example{
   public static void main(String args[]){
        ABC obj = new ABC();
        System.out.println(obj.num);
        System.out.println(obj.square(10));
   }
}
```

EXERCISES. True or False. Mark the following statements as True or False. Write your answer on the space provided before each number. (2 points each)

_____1. Private modifier is not limited to class only.

_____2. Class and Interface can be declared as private.

_____3. If a class has private constructor then you cannot create the object of that class from outside of the class.

_____4. Private variables can be accessed outside the class.

_____5. Private methods are only accessible within a class.

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Java Access Modifier: Public Access Modifier		
Lesson Competency: Use public access modifier.		
References : https://beginnersbook.com/2013/05/java-access-modifiers/		LAS No.: 48

CONCEPT NOTES

Public access modifier

The members, methods and classes that are declared public can be accessed from anywhere. This modifier doesn't put any restriction on the access.

Sample Program

Addition.java

```
package abcpackage;
public class Addition {
  public int addTwoNumbers(int a, int b){
      return a+b;
  }
}
```

Test.java

```
package xyzpackage;
import abcpackage.*;
class Test{
  public static void main(String args[]){
    Addition obj = new Addition();
    System.out.println(obj.addTwoNumbers(100, 1));
  }
}
```

Output: 101

EXERCISES. Perform the following. (20 points)

1. Using the above programs, modify the **Addition.java** class by defining a new public method named addThreeNumbers() that holds 3 parameters of type int. Your method should return a value of type int. Call the new method inside main() supplying the values 8,17 and 5 respectively.

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Java Access Modifier: Protected Access Modifier		
Lesson Competency: Identify the usage of protected access modifier.		
References : https://beginnersbook.com/2013/05/java-access-modifiers/		LAS No.: 49

CONCEPT NOTES

Protected Access Modifier

Protected data member and method are only accessible by the classes of the same package and the subclasses present in any package. Classes cannot be declared protected. This access modifier is generally used in a parent child relationship.

Sample Program

In this example the class Test which is present in another package is able to call the addTwoNumbers() method, which is declared protected. This is because the Test class extends class Addition and the protected modifier allows the access of protected members in subclasses (in any packages).

Addition.java

```
package abcpackage;
public class Addition {

  protected int addTwoNumbers(int a, int b){
      return a+b;
  }
}
```

Test.java

```
package xyzpackage;
import abcpackage.*;
class Test extends Addition{
  public static void main(String args[]){
      Test obj = new Test();
      System.out.println(obj.addTwoNumbers(11, 22));
  }
}
```

Output: 33

EXERCISES. Answer the following question. (5 points)

1. Using the sample program above, what is the use of making a method protected?

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Java Type Casting: Widening		
Lesson Competency: Trace program outputs that uses type casting.		
References : https://www.w3schools.com/java/java_type_casting.asp		LAS No.: 50

CONCEPT NOTES

Java Type Casting

Type casting is when you assign a value of one primitive data type to another type.

Two types of casting:

- **Widening Casting** (automatically) - converting a smaller type to a larger type size
  byte -> short -> char -> int -> long -> float -> double
- **Narrowing Casting** (manually) - converting a larger type to a smaller size type
  double -> float -> long -> int -> char -> short -> byte

Widening Casting Sample Program

```
public class MyClass {
  public static void main(String[] args) {
    int myInt = 9;
    double myDouble = myInt;          // Automatic casting: int to double
    System.out.println(myInt);        // Outputs 9
    System.out.println(myDouble);     // Outputs 9.0
  }
}
```

EXERCISES. Program Tracing. Identify the output of the following program. (10 points)

```
public class MyClass {
  public static void main(String[] args) {
    int num1 = 12, num2 = 5;
    double newNum1 = num1, newNum2 = num2;
    System.out.println(num1);
    System.out.println(num2);
    System.out.println(newNum1);
   System.out.println(newNum2);
    System.out.println((num1 + newNum2));
  }
}
```

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Java Type Casting: Narrowing		
Lesson Competency: Trace program outputs that uses type casting.		
References : https://www.w3schools.com/java/java_type_casting.asp		LAS No.: 51

CONCEPT NOTES

Narrowing Casting -must be done manually by placing the type in parentheses in front of the value:

Syntax: `dataType variableName = (dataType) variableToConvert;`

Sample Program

```
public class MyClass {
  public static void main(String[] args) {
    double myDouble = 9.78;
    int myInt = (int) myDouble;          // Manual casting: double to int

    System.out.println(myDouble);        // Outputs 9.78
    System.out.println(myInt);           // Outputs 9
  }
}
```

EXERCISES. Program Tracing. Identify the output of the following program: (10 points)

```
public class MyClass {
  public static void main(String[] args) {
    double num1 = 12.85 num2 = 5.41;
    int newNum1 = (int) num1;
    int newNum2 = (int) num2;

    System.out.println(newNum1);
    System.out.println(newNum2);
    System.out.println((newNum1 + newNum2));
    System.out.println((num1 + newNum2));
  }
}
```

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Java Virtual Methods		
Lesson Competency: Use virtual method invocation in accordance with Java framework		
References : https://www.thejavaprogrammer.com/java-virtual-method/		LAS No.: 52

CONCEPT NOTES

Java Virtual Method

When a Super class reference holds a subclass method, calling the same method using the super class reference will automatically call the subclass method.

In an object oriented java programming, instead of the reference, we are calling the method in accordance to its object. The methods which are implementing this concept are the **java virtual methods**.

The prerequisites of a virtual function are:

- Inheritance
- Polymorphism

```
class Coach {
	//virtual method
	void per() {
		System.out.println("Coach Class");
	}
}
 public class Sportscoach extends Coach {
	void per() {
		System.out.println("Sportscoach Class");
	}
	public static void main (String args[]) {
		Coach obj = new Sportscoach();
		obj.per();
	}
}
```

Output:

Sportscoach Class

EXERCISES. Perform the following: (15 points)

1. Revise the program above such that the statement under the virtual method will be printed on the screen.

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Method Overriding		
Lesson Competency: Identify the rules in implementing method overriding.		
References : https://www.w3schools.in/java-tutorial/method-overriding/		LAS No.: 53

CONCEPT NOTES

Method Overriding refers to declaring a method in the subclass which already exists in the parent class. When a class is inheriting a method from a superclass of its own, then there is an option of overriding the method provided it is not declared as final.

The advantage of using overriding is the ability to classify a behavior that's specific to the child class, and the child class can implement a parent class method based on its necessity.

The following are the rules that a programmer should follow to implement overriding.

1. In Java, a method can only be written in the child class and not in same class.
2. Argument list should be the same as that of the overridden method of that class.
3. Instance methods can also be overridden if they are inherited by the child class.
4. A constructor cannot be overridden.
5. Final - declared methods cannot be overridden.
6. Any method that is static cannot be used to override.
7. The return type must have to be the same, or a subtype of the return type declared in the original overridden method in the parent class.
8. If a method cannot be inherited, then it cannot be overridden.
9. A child class within the same package as the instance's parent class can override any parent class method that is not declared private or final.
10. A child class in a different package can only override the non-final methods declared as public or protected.

EXERCISES. True or False. Mark the following as true or False. (2 points each)

_____1. A constructor can be overriden.

_____2. Any method that is static can be overriden.

_____3. If a method cannot be inherited, then it can be overridden.

_____4. Argument list should not be the same as that of the overridden method of that class.

_____5. Methods declared as final cannot be overridden.

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Method Overriding: A Program		
Lesson Competency: Trace program outputs that uses Method Overriding.		
References : https://www.w3schools.in/java-tutorial/method-overriding/		LAS No.: 54

CONCEPT NOTES

A real example of Java Method Overriding

Consider a scenario where Bank is a class that provides functionality to get the rate of interest. However, the rate of interest varies according to banks. For example BDO, BPI and DBP banks could provide 8%, 7%, and 9% rate of interest.

```
//Creating a parent class.
class Bank{
    int getRateOfInterest(){return 0;}
}
//Creating child classes.
class BDO extends Bank{
    int getRateOfInterest(){return 9;}
}
class BPI extends Bank{
    int getRateOfInterest(){return 8;}
}
class DBP extends Bank{
    int getRateOfInterest(){return 6;}
}
//Test class to create objects and call the methods
class Test2{
public static void main(String args[]){
    BDO s=new BDO();
    BPI i=new BPI();
    DBP a=new DBP();
    System.out.println("BDO Rate of Interest: "+s.getRateOfInterest());
    System.out.println("BPI Rate of Interest: "+i.getRateOfInterest());
    System.out.println("DBP Rate of Interest: "+a.getRateOfInterest());
    }
}
```

EXERCISE. Program Tracing. Trace the above program in order to identify what is printed on the screen if we will run it on a Java Compiler. (20 points)

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Package and import statements and its uses		
Lesson Competency: Justify the use of package and import statements in accordance with Java framework		
References : https://beginnersbook.com/2013/03/packages-in-java/		LAS No.: 55

CONCEPT NOTES

A **package** as the name suggests is a pack(group) of classes, interfaces and other packages. In java we use packages to organize our classes and interfaces.

2 types of packages in Java:

1. **built-in packages** - already defined package like java.io.*, java.lang.* etc
2. **user defined** package – the package we create

In java we have several built-in packages, for example when we need user input, we import a package like this:

```
import java.util.Scanner;
```

Here:

→ **java** is a top level package

→ **util** is a sub package

→ and **Scanner** is a class which is present in the sub package **util**.

Reasons why you should use packages in Java:

- **Reusability**: While developing a project in java, we often feel that there are few things that we are writing again and again in our code. Using packages, you can create such things in form of classes inside a package and whenever you need to perform that same task, just import that package and use the class.
- **Better Organization**: Again, in large java projects where we have several hundreds of classes, it is always required to group the similar types of classes in a meaningful package name so that you can organize your project better and when you need something you can quickly locate it and use it, which improves the efficiency.
- **Name Conflicts**: We can define two classes with the same name in different packages so to avoid name collision.

EXERCISES. Discussion. Answer the following question. (10 points)

1. Discuss the reasons why there is a need to use packages and import statements in Java.

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Package and import statements and its uses: A Program		
Lesson Competency: Use package and import statements in accordance with Java framework		
References : https://beginnersbook.com/2013/03/packages-in-java/		LAS No.: 56

CONCEPT NOTES

Java packages

To create a class inside a package, declare the package name in the first statement in your program. A class can have only one package declaration.

Calculator.java file created inside a package letmecalculate

```
package letmecalculate;
public class Calculator {
  public int add(int a, int b){
      return a+b;
  }
  public static void main(String args[]){
      Calculator obj = new Calculator();
      System.out.println(obj.add(10, 20));
  }
}
```

Now lets see how to use this package in another program.

```
import letmecalculate.Calculator;
public class Demo{
  public static void main(String args[]){
      Calculator obj = new Calculator();
      System.out.println(obj.add(100, 200));
  }
}
```

To use the class Calculator, import the package letmecalculate. However if you have several classes inside package letmecalculate then you can import the package like this, to use all the classes of this package.

```
import letmecalculate.*;
```

EXERCISES. Perform the following. (20 points)

1. Revise the Demo class above by allowing the user to enter the values to be added. You can do it by importing the Scanner class under java.util package.

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Java Abstract Classes: Its Uses		
Lesson Competency: Discuss the usage of abstract classes		
References : https://beginnersbook.com/2013/05/java-abstract-class-method/ https://www.w3schools.com/java/java_abstract.asp		LAS No.: 57

CONCEPT NOTES

Data abstraction is the process of hiding certain details and showing only essential information to the user. It can be achieved with either **abstract classes** or **interfaces**

The abstract keyword is a non-access modifier, used for classes and methods:

- **Abstract class:** is a restricted class that cannot be used to create objects (to access it, it must be inherited from another class).
- **Abstract method:** can only be used in an abstract class, and it does not have a body. The body is provided by the subclass (inherited from).

An **abstract class** is mostly used to provide a base for subclasses to extend and implement the abstract methods and override or use the implemented methods in abstract class.

Why we need an abstract class?

Lets say we have a class Animal that has a method sound() and the subclasses of it like Cat, Dog, Lion, Horse etc. Since the animal sound differs from one animal to another, there is no point to implement this method in parent class. This is because every child class must override this method to give its own implementation details, like Lion class will say "Roar" in this method and Dogclass will say "Woof".

So when we know that all the animal child classes will and should override this method, then there is no point to implement this method in parent class. Thus, making this method abstract would be the good choice as by making this method abstract we force all the sub classes to implement this method(otherwise you will get compilation error), also we need not to give any implementation to this method in parent class.

EXERCISES Answer the following question.

1. Discuss the usage of abstract classes. (10 pts)

DepED DEPARTMENT OF EDUCATION K TO 12 PROGRAM

SHARED OPTIONS

SENIOR HIGH ALTERNATIVE RESPONSIVE EDUCATION DELIVERY

GRADE 11 DLP LEARNING ACTIVITY SHEET

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Static and Final Keywords		
Lesson Competency: Identify the usage of static and final keywords in accordance with Java framework		
References : https://www.w3schools.com/java/ref_keyword_static.asp		LAS No.: 58

CONCEPT NOTES

A **static method** means that it can be accessed without creating an object of the class, unlike public.

Sample Program:

```
public class MyClass {
  static void myStaticMethod() {   // Static method
    System.out.println("Static methods can be called without creating objects");
  }
  public void myPublicMethod() {   // Public method
    System.out.println("Public methods must be called by creating objects");
  }
  // Main method
  public static void main(String[ ] args) {
    myStaticMethod(); // Call the static method
    // myPublicMethod(); This would output an error
    MyClass myObj = new MyClass();      // Create an object of MyClass
    myObj.myPublicMethod();             // Call the public method
  }}
```

Set a variable to final, to prevent it from being overridden/modified:

Sample Program:

```
public class MyClass {
  final int x = 10;
  public static void main(String[] args) {
    MyClass myObj = new MyClass();
    myObj.x = 25; // will generate an error: cannot assign a value to a final variable
    System.out.println(myObj.x);
  }
}
```

EXERCISES. Discuss the following: (5 points each)

1. When to use static keyword?
2. When to use final keyword?

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Nested Classes		
Lesson Competency: Trace output of the nested classes in accordance with Java framework		
References : https://www.w3schools.com/java/java_inner_classes.asp		LAS No.: 59

CONCEPT NOTES

Nested Classes- It is possible to nest classes (a class within a class) in Java. The purpose of nested classes is to group classes that belong together, which makes your code more readable and maintainable. To access the inner class, create an object of the outer class, and then create an object of the inner class:

Sample Program:

```
class OuterClass {
  int x = 20;
  class InnerClass {
    int y = 10;
  }}
public class MyMainClass {
  public static void main(String[] args) {
    OuterClass myOuter = new OuterClass();
    OuterClass.InnerClass myInner = myOuter.new InnerClass();
    System.out.println((myInner.y + myOuter.x));
  }}
```

Output: 30

EXERCISES. Program Tracing. What is written to the standard output as the result of executing the following code? (15 points)

```
public class Outer {
     private int a = 3;
     Outer(){
       a = 8; }
    class Nested {
        Nested() {
            System.out.print("-a"+a);
            a = 4;
        }}
    public static void main(String[] args) {
        Nested nested = new Outer().new Nested();
    }
```

}

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Using enumerated types		
Lesson Competency: Use enumerated types in accordance with Java framework		
References : https://www.geeksforgeeks.org/enum-in-java/		LAS No.: 60

CONCEPT NOTES

enum in Java - Enumerations serve the purpose of representing a group of named constants in a programming language.

- are used when we know all possible values at **compile time**, such as choices on a menu, rounding modes, and command line flags.

Example:

```
enum Level {
  LOW, MEDIUM, HIGH;
}
```

You can access enum constants with the dot syntax:

```
Level myVar = Level.MEDIUM;
```

Sample Program:

```
enum Level {
  LOW,MEDIUM,HIGH;
}
public class MyClass {
  public static void main(String[] args) {
    Level myVar = Level.MEDIUM;
    switch(myVar) {
      case LOW:
        System.out.println("Low level"); break;
      case MEDIUM:
         System.out.println("Medium level"); break;
      case HIGH:
        System.out.println("High level"); break;
    }
  }
}
```

EXERCISES: Perform the following: (15 points)

1. Using the program above, write a code to retrieve all the values of enum and display it on the screen.

DepED DEPARTMENT OF EDUCATION K TO 12 PROGRAM

SHARED OPTIONS
SENIOR HIGH ALTERNATIVE RESPONSIVE EDUCATION DELIVERY
GRADE 11 DLP LEARNING ACTIVITY SHEET

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Interfaces		
Lesson Competency: Identify the characteristics of interfaces in Java		
References : https://www.geeksforgeeks.org/interfaces-in-java/		LAS No.: 61

CONCEPT NOTES

Interfaces in Java

Like a class, an interface can have methods and variables, but the methods declared in interface are by default abstract (only method signature, no body).

Syntax :

```
interface <interface_name> {
   // declare constant fields
   // declare methods that abstract by default.
}
```

To declare an interface, use **interface** keyword. It is used to provide total abstraction. That means all the methods in interface are declared with empty body and are public and all fields are public, static and final by default. A class that implement interface must implement all the methods declared in the interface.

Why do we use interface?

- It is used to achieve total abstraction.
- Since java does not support multiple inheritance in case of class, but by using interface it can achieve multiple inheritance.
- It is also used to achieve loose coupling.
- Interfaces are used to implement abstraction.

```
// A simple interface
interface Player
{
    final int id = 10;
    int move();
}
```

EXERCISES. True or False. Mark the following as True or False. (2 pts each)

_____1. Interface cannot contain methods.

_____2. Interface is used to implement abstraction.

_____3. A class that implement interface may not implement all the methods declared in the interface.

_____4. All the methods in interface are declared with empty body.

_____5. Java class support multiple inheritance.

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Implement Interfaces		
Lesson Competency: Write code that declares, implements, and/or extends interfaces		
References : https://www.geeksforgeeks.org/interfaces-in-java/ http://www.java-programming.info/tutorial/pdf/java/exercises/exercises-java-8-interfaces.pdf		LAS No.: 62

CONCEPT NOTES

Implement Interfaces- To implement an interface we use keyword: **implement**

Java program to demonstrate working of interface.

```
import java.io.*;
// A simple interface
interface in1
{
    // public, static and final
    final int a = 20;

    // public and abstract
    void display();
}
// A class that implements interface.
class testClass implements in1
{
    // Implementing the capabilities of
    // interface.
    public void display()
    {
        System.out.println("Welcome!");
    }
    // Test Case
    public static void main (String[] args)
    {
        testClass t = new testClass();
        t.display();
        System.out.println(a);
    }
}
```

Output: Welcome
20

EXERCISES. Perform the following: (50 points)

1. Start by making a normal Java application with these features:
   - An interface called RegularPolygon with two abstract methods: getNumSides and getSideLength.
   - A class EquilateralTriangle that implements the interface, has getNumSides return 4 and getSideLength return an instance variable that is set by the constructor.
   - A class Square that implements the interface, has getNumSides return 5 and getSideLength return an instance variable that is set by the constructor.
2. Add a static totalSides method, that given a RegularPolygon[], returns the sum of the number of sides of all the elements.
3. Add two default methods:
   - getPerimeter (n * length, where n is the number of sides)
   - getInteriorAngle ((n-2)π/n in radians)

4. Make a few test cases.

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Java Generics		
Lesson Competency: Create Java Generic Arrays in accordance with Java framework		
References : https://www.studytonight.com/java/generic-in-java.php		LAS No.: 63

CONCEPT NOTES

Java Generics

Generic programming allows the programmer to create classes, interfaces and methods in which type of data is specified as a parameter. It offers a facility to write an algorithm independent of any specific type of data. Generics also provide type safety. Type safety means ensuring that an operation is being performed on the right type of data before executing that operation.

Using Generics, it has become possible to create a single class, interface or method that automatically works with all types of data(Integer, String, Float etc). It has expanded the ability to reuse the code safely and easily.

Syntax for creating an object of a generic type

Class_name <data type> reference_name = new Class_name<data type> ();
or
Class_name <data type> reference_name = new Class_name<>();

Example:

```
List<Integer> list = new ArrayList<Integer>();
 list.add(1000);     //works fine
 list.add("lokesh"); //compile time error;
```

EXERCISES. Answer the following: (5 points each)

1. Create an ArrayList named listOfScores of type double.

2. Add three elements in your ArrayList, 7, 8 and 3 respectively.

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Using throw statements		
Lesson Competency: Use Throw statements in accordance with Java framework		
References : https://www.w3schools.com/java/ref_keyword_throw.asp		LAS No.: 64

CONCEPT NOTES

Using throw statement

throw keyword - is used to create a custom error. It is used together with an **exception type**. There are many exception types available Java: ArithmeticException, ClassNotFoundException, SecurityException, ArrayIndexOutOfBoundsException, etc.

Example:

Throw an exception if **age** is below 18 (print "Access denied"). If age is 18 or older, print "Access granted":

```
public class MyClass {
  static void checkAge(int age) {
    if (age < 18) {
      throw new ArithmeticException("Access denied - You must be at least 18 years old.");
    }
    else {
      System.out.println("Access granted - You are old enough!");
    }
  }
  public static void main(String[] args) {
    checkAge(15);        // Set age to 15 (which is below 18...)
  }
}
```

EXERCISES. Perform the following: (15 points)

Revise the above program such that it will also throw an exception if the age set is 130 and above (print "Access denied – Invalid age").

DepED DEPARTMENT OF EDUCATION K TO 12 PROGRAM

SHARED OPTIONS

SENIOR HIGH ALTERNATIVE RESPONSIVE EDUCATION DELIVERY

GRADE 11 DLP LEARNING ACTIVITY SHEET

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : JDBC API		
Lesson Competency: Identify the layout of the JDBC API in accordance with Java framework		
References : https://www.wisdomjobs.com/e-university/jdbc-tutorial-278/what-is-jdbc-api-1107.html		LAS No.: 65

CONCEPT NOTES

JDBC (Java Database Connectivity) is a set of programming APIs (Application Package Interface) that allows easy connection to a wide range of databases (especially relational databases) through Java programs.

java.sql provides the API for accessing and processing data stored in a data source(usually a relational database) using the Java programming language. This package provides the foundation and most commonly used objects(such as **Connection**, **ResultSet**, **Statement**, and **PreparedStatement**). Also, this package provides classes and interfaces to get both database and result set metadata from the database server. This package has a set of classes and interfaces (such as DatabaseMetaData and ResultSetMetaData) that deal with database metadata, which will be one of the focuses of this book.

javax.sql provides the API for server-side data source access.

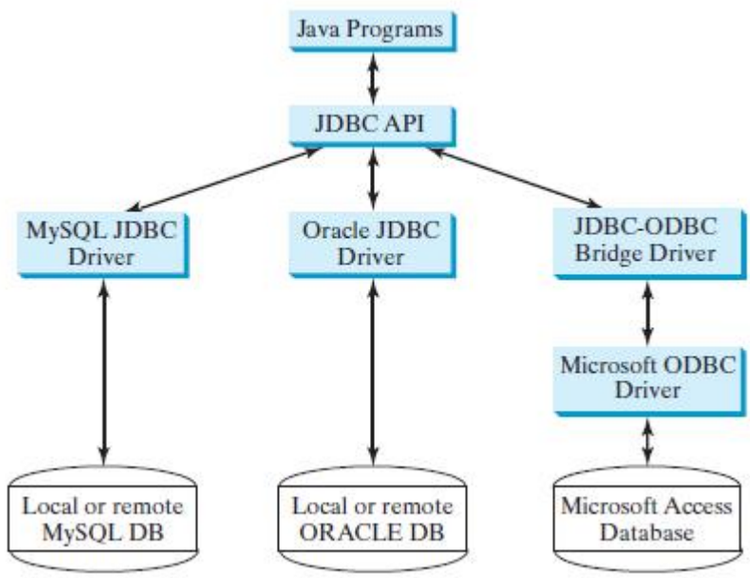


Figure shows how a database application uses JDBC to interact with one or more databases.

EXERCISES.

I. Answer the following questions. Write your answer on the space provided before each number.

_____1. It refers to set of programming APIs that allows easy connection to a wide range of databases. (2 pts)

_____2. It provides the API for server-side data source access. (2 pts)

_____3. It provides classes and interfaces to get both database and result set metadata from the database server. (2 pts)

_____4. What are the objects that can be found in java.sql package? (4 pts)

II. What is the purpose of JDBC? (5 pts)

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Using JDBC driver to connect to database		
Lesson Competency: Discuss the procedure in using JDBC driver to connect to database in accordance with Java framework		
References : https://www.javatpoint.com/steps-to-connect-to-the-database-in-java		LAS No.: 66

CONCEPT NOTES

5 steps to connect any java application with a database using JDBC

1. Register the Driver class - The **forName()** method of Class class is used to register the driver class. This method is used to dynamically load the driver class.

```
Class.forName("JDBCDriverClass");
```

2. Create the connection object - The **getConnection()** method of DriverManager class is used to establish connection with the database.

```
Connection con =
DriverManager.getConnection( "jdbc:mysql://hostname/dbname","username",
"password");
```

3. Create statement - The **createStatement**() method of Connection interface is used to create statement. The object of statement is responsible to execute queries with the database.

```
Statement stmt=con.createStatement();
```

4. Execute queries - The **executeQuery()** method of Statement interface is used to execute queries to the database. This method returns the object of ResultSet that can be used to get all the records of a table.

```
ResultSet rs=stmt.executeQuery("select * from studentTable");
while(rs.next()){
System.out.println(rs.getInt(1)+" "+rs.getString(2));
}
```

5. Close connection - By closing connection object statement and ResultSet will be closed automatically. The **close()** method of Connection interface is used to close the connection.

```
con.close();
```

EXERCISES. Answer the following question: (10 points)

1. Discuss the procedure in connecting a Java application with a database using JDBC.

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Using JDBC driver to connect to database		
Lesson Competency: Use JDBC driver to connect to database in accordance with Java framework		
References : https://www.javatpoint.com/steps-to-connect-to-the-database-in-java		LAS No.: 67

CONCEPT NOTES

A Sample Program that uses JDBC driver to connect to a database

```
import java.sql.*;

public class SimpleJdbc {
 public static void main(String[] args)
 throws SQLException, ClassNotFoundException {

    Class.forName("com.mysql.jdbc.Driver");
    System.out.println("Driver loaded");

    Connection connection = DriverManager.getConnection
    ("jdbc:mysql://localhost/studentdb" , "root", "1234");
    System.out.println("Database connected");

    Statement statement = connection.createStatement();

    ResultSet resultSet = statement.executeQuery
    ("select firstName, mi, lastName from StudentTable where lastName "
    + " = 'Macaron'");

    while (resultSet.next())
        System.out.println(resultSet.getString(1) + "\t" +
        resultSet.getString(2) + "\t" + resultSet.getString(3));

    connection.close();
   }
 }
```

EXERCISES. Study the program above and perform the following: (30 pts)
Given a database named studentdb (no password) that contains a table named ScoreTable having the following fields studentID, score, and subjectName, create a program that prints the content of the ScoreTable. Save your program as **MyFirstJDBCProgram.java.**

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Applying JDBC Row Set Provider, Row Set Factory, and Row Set Interfaces		
Lesson Competency: Use JDBC Row Set Provider, Row Set Factory, and Row Set interfaces in Java Framework		
References : https://www.javatpoint.com/jdbc-rowset		LAS No.: 68

CONCEPT NOTES

JDBC RowSet

The instance of **RowSet** is the java bean component because it has properties and java bean notification mechanism. It is introduced since JDK 5. It is the wrapper of ResultSet.

Sample Program:

```
import java.sql.*;
import javax.sql.RowSetEvent;
import javax.sql.RowSetListener;
import javax.sql.rowset.JdbcRowSet;
import javax.sql.rowset.RowSetProvider;
public class RowSetImplementation {
      public static void main(String[] args) throws Exception {
            Class.forName("JDBCDriverClass ");
   //Creating and Executing RowSet
      JdbcRowSet rowSet = RowSetProvider.newFactory().createJdbcRowSet();
      rowSet.setUrl("jdbc:mysql://localhost/dbname ");  //use your database name
      rowSet.setUsername("root");
      rowSet.setPassword("1234");
      rowSet.setCommand("select * from employeeTable");
      rowSet.execute();
   while (rowSet.next()) {
                  System.out.println("Id: " + rowSet.getString(1));
                  System.out.println("Name: " + rowSet.getString(2));
                  System.out.println("Salary: " + rowSet.getString(3));
           }
} }
```

EXERCISES. Perform the following: (30 points).

Run the above program but make use of your own database. In your database, create a table named employeeTable that contains the fields id, name, and salary. Save at least three data in your tale and display the result using JDBC RowSet. Save your program as **RowSet_YourLastName.java.**

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Creating and using PreparedStatement		
Lesson Competency: Identify the methods used by PreparedStatement		
References : https://www.javatpoint.com/PreparedStatement-interface		LAS No.: 69

CONCEPT NOTES

PreparedStatement interface is a sub interface of Statement. It is used to execute parameterized query.

Example of parameterized query:

String sql="insert into emp values(?,?,?)";

We are passing parameter (?) for the values. Its value will be set by calling the setter methods of PreparedStatement.

Why use PreparedStatement?

Improves performance: The performance of the application will be faster if you use PreparedStatement interface because query is compiled only once.

Methods of PreparedStatement interface

The important methods of PreparedStatement interface are given below:

Method	Description
public void setInt(int paramIndex, int value)	sets the integer value to the given parameter index.
public void setString(int paramIndex, String value)	sets the String value to the given parameter index.
public void setFloat(int paramIndex, float value)	sets the float value to the given parameter index.
public void setDouble(int paramIndex, double value)	sets the double value to the given parameter index.
public int executeUpdate()	executes the query. It is used for create, drop, insert, update, delete etc.
public ResultSet executeQuery()	executes the select query. It returns an instance of ResultSet.

EXERCISES. Identify the following: (2 points each)

_____1. It is used to execute parameterized query.

_____2. It is a method that sets the String value to the given parameter index.

_____3. It is a method that sets the double value to the given parameter index.

_____4. It is a method that executes the select query.

_____5. What value is being set by setFloat() method?

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Creating and using PreparedStatement- Insert Record		
Lesson Competency: Insert records using a program that uses PreparedStatement		
References : https://www.javatpoint.com/PreparedStatement-interface		LAS No.: 70

CONCEPT NOTES

PreparedStatement Interface that Inserts a Record

Use the table as given below:

create table employeeTable(id number(10),name varchar2(50), salary number(10));

Insert records in this table using the code given below:

```
import java.sql.*;
class InsertPreparedStatement{
public static void main(String args[]){
    try{
    Class.forName("JDBCDriverClass ");

    Connection
    con=DriverManager.getConnection"jdbc:mysql://hostname/dbname","usernam
    e","password);

    PreparedStatement stmt=con.prepareStatement("insert into employeeTable
    values(?,?)");
    stmt.setInt(1,101);//1 specifies the first parameter in the query
    stmt.setString(2,"Mara");
    stmt.setDouble(3,27000);

    int i=stmt.executeUpdate();
    System.out.println(i+" Successfully inserted records");
    con.close();

    }catch(Exception e){ System.out.println(e);
    }
}
}
```

EXERCISES. Perform the following: (30 points)

Run the above program, make sure to create table **employeeTable** first. Under Connection statement, use your own hostname, database name, username and password. Then, insert at least 10 records in your table using insert PreparedStatement.

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Creating and using PreparedStatement- Update Record		
Lesson Competency: Create and use PreparedStatement		
References : https://www.javatpoint.com/PreparedStatement-interface		LAS No.: 71

CONCEPT NOTES

PreparedStatement Interface that Updates a Record

Use the codes given below to update records of the table **employeeTable** use in the previous activity:

```
import java.sql.*;
class UpdatePreparedStatement{
public static void main(String args[]){
        try{
        Class.forName("JDBCDriverClass ");

        Connection
        con=DriverManager.getConnection"jdbc:mysql://hostname/dbname","usernam
        e","password);

        PreparedStatement stmt=con.prepareStatement("update employeeTable set
        name=? where id=?");
        stmt.setString(1,"Maricar");//1 specifies the first parameter in the query
        i.e. name
        stmt.setInt(2,101);  //look for id that is found in your table

        int i=stmt.executeUpdate();
        System.out.println(i+" Successfully updated records");

        con.close();

        }catch(Exception e){ System.out.println(e);
        }
  }
}
```

EXERCISES. Performance Task. Perform the following: (30 points)
Run the above program. Use the database in your previous activity. Then, update the salary of the first three employees by adding P 500.00 to their salary using update PreparedStatement.

Name:	Date:	Score:
Subject : TVL-COMPUTER PROGRAMMING (JAVA) NC III		
Lesson Title : Creating and using PreparedStatement- Delete Record		
Lesson Competency: Create and use PreparedStatement		
References : https://www.javatpoint.com/PreparedStatement-interface		LAS No.: 72

CONCEPT NOTES

PreparedStatement Interface that Deletes a Record

```
import java.sql.*;
class DeletePreparedStatement{
public static void main(String args[]){
        try{
        Class.forName("JDBCDriverClass ");

        Connection
        con=DriverManager.getConnection"jdbc:mysql://hostname/dbname","usernam
        e","password);

        PreparedStatement stmt=con.prepareStatement("delete from
        employeeTable  where id=?");
        stmt.setInt(1,101);


        int i=stmt.executeUpdate();
        System.out.println(i+" Successfully deleted record");
        con.close();

        }catch(Exception e){ System.out.println(e);
        }
}
}
```

EXERCISES. Performance Task. Perform the following: (30 points)
Run the above program. Use the database in your previous activity. Then, delete the data in your table where salary is less than 20000 using delete PreparedStatement