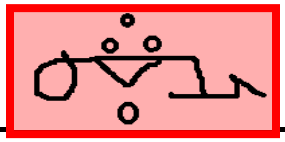
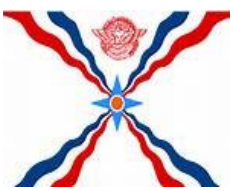




Wright College + Chapter 3

Object-Oriented Programming (OOP): Class, Object, Encapsulation, Polymorphism, Inheritance, and Abstraction

CIS 144 Java Programming Language– Introduction to Computer Programming

"Hands-On" Mastering Computer Logic, Design and Programming Using Java Language

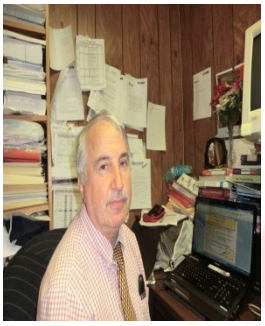
Written By:

Ogar Haji

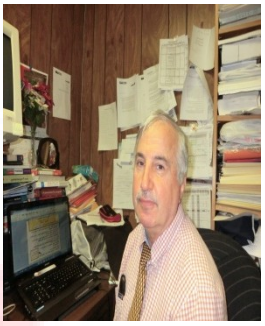
MS Computer Science

DePaul University ✚ Chicago, Illinois

Date Published: February 5, 2021

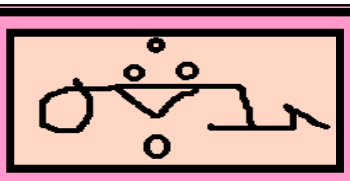


Computer Text Books Published by the Author: Ogar Haji

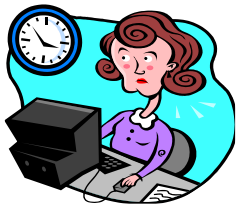


The Following is a List of Computer Text Books Published by the Author: Ogar Haji. He has an MS Degree in Computer Science from DePaul University, Chicago, Illinois - USA. Mr. Ogar Haji has over 30 Years of teaching experience at: The College of Office Technology, Oakton College, Washington College, Truman College, Wright College, Triton College, ITT Technical Institute, Phoenix University and East+West University in Chicago, Illinois.

1) **"Hands-On" Mastering Microsoft Windows 7 and Vista**
2) **"Hands-On" Mastering Microsoft Excel 2010 and 2007**
3) **"Hands-On" Mastering Microsoft Word 2010 and 2007**
4) **"Hands-On" Mastering Microsoft Access 2010 and 2007**
5) **"Hands-On" Mastering Microsoft PowerPoint 2010 & 2007**
6) **"Hands-On" Mastering Microsoft Publisher 2010**
7) **"Hands-On" Mastering MS Visual Basic .Net Language**
8) **"Hands-On" Mastering Java Programming Language**
9) **"Hands-On" Mastering Html5 and CSS3 Web Page Design**
10) **"Hands-On" Mastering JavaScript Programming Language**
11) **"Hands-On" Mastering Ruby Programming Language**
12) **"Hands-On" Mastering Python Programming Language**
13) **"Hands-On" Mastering QBasic Programming Language**
14) **"Hands-On" Mastering DOS (Disk Operating System)**
15) **"Hands-On" Mastering C# Programming Language**



Ogar Haji (Computer Instructor)
Harry S. Truman College
1145 West Wilson Avenue
Chicago, Illinois 60640
USA



Computer Labs Rules

1- **No Drinks, Food, Headphones** allowed in Computer labs. And Please **Turn Off the Cell Phones**.

2- When **Lecturing** is in progress, you are Not allowed to work on the computer. Please **Pay Attention** and **Take Notes**.

3- **Attendance and Punctuality** are very **important**. If you are absent, it is your responsibility to make up for the missing work and assignments. **Attendance will be taken daily.**

4- Students should have a **USB Flash Drive** and **Save Projects** to it.

5- **Practice makes perfect**. Please keep practicing the new features or steps over and over again until the instructor tells you to stop.

6- You have to **Concentrate** on what you are doing. **Talking is Not Allowed** in the computer Lab.

7- Please **Study** the Lessons in your **Java Handout and Text Book Daily** and **review your notes before class. There will be a Quiz Once a Week.**

8- Please **Check Mark ☑ the Lessons** in the Handout that you **have completed.**

9- You **must do All Java works**, **Assignments and Tests** located at the End of each Chapter on Time.

CIS144 Java Programming
Instructor:
Ogar Haji

Chapter 3

Object-Oriented Programming (OOP): Class, Object, Encapsulation, Polymorphism, Inheritance, and Abstraction

You will learn the following in Chapter 3:

- **Working with NetBeans IDE (Windows Forms)**
- **Start a New Windows JFrame Forms Application in Java**
- **Understanding NetBeans IDE Java screen**
- **Display Properties Windows on the screen**
- **Create User Interface Forms & Change Text and Name Properties**
- **Create Calculate Average User Interface Form**
- **Doing Math on 2 Numbers (Add, Subtract, Multiply, Divide)**
- **Calculate Gross Pay Project GUI program**
- **Insert Today's Date and Current Time – Lab Exercise**
- **Show and Hide a Picture – Lab Exercise**
- **Modify Gross Pay Project – Add Show/Hide Picture**
- **Convert from Celsius temperature to Fahrenheit temp**
- **Do Java Chapter 3 Homework #3**
- **Do Java Lab Assignments 3**

❖

Input/Output

Flowchart Symbols

Processing

+++ Review +++

Lesson 70 Review : What are the Flowchart Symbols used in Java Language?

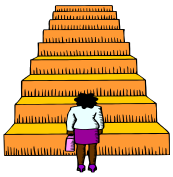
You should always **draw a Flowchart** when you Design, Code and Solve a problem in Java language.

Before you Code a program in Java Language, you have to **Draw a Flowchart** to solve the problem of the program you want to code.

The following symbols are used with Java Programming Language:

Symbol	Symbol Name	Usage
Start or End	**Oval (Beginning and Terminal)** symbol	Use **Oval (Beginning and Terminal) Symbol** at the Beginning of the Flowchart and at the End of the Flowchart. Use with Start and End statements.
Print and Read	**Parallelogram (Input/Output)** Symbol	Use **Parallelogram (Input /Output)** or **I/O** symbol to Input Data, Read Input or Print Output
grossPay = hours * rate	**Rectangle** Symbol	Use the **Rectangle** Symbol for Calculating, Assigning Values
↓ →	**FlowLine** Symbol	Use **FlowLine** Symbols to show the Flow or Sequence of the flowchart.
if(hours>40)	**Diamond (Decision)** Symbol	Use **Diamond (Decision)** Symbol with the If or Select statements when deciding if Hours is > 40. The Result will be either True or False.
A	**Connector** Symbol	Use **Connector** Symbol to Connect the Flowchart rather than draw a long Arrow. Use A

calculateGrossPay()	**Function** or **Method (Predefined Process)** Symbol.	Use **Function** or **Method (Predefined Process)** Symbol to call another Function or Method that contains coding statements.



Calculate Gross Pay of Employees Project

+++ (Do Lab Exercise) +++

+++ Review +++

Lesson 71 Ex : How to Calculate Gross Pay of Employees Project?

Problem or Project: Design and Code in Java Language the project to Calculate Gross Pay of Employees in a company.

Do the following 12 Must Steps to Design, Code and Solve a project using Java Language.

Do Steps 1 thru 7 in your Note Book or on Paper.

Step 1) Purpose of the Program: State what Program will do: (5 Points)

a) This Program will calculate Gross Pay of Employees.
b) It will ask the User to Enter Employee's Full Name:
c) It will ask the User to Enter Number of Hours Worked and
d) It will ask the User to Enter Hourly Rate.
e) The program will then calculate Gross Pay.

Gross Pay = Hours * Rate

f) Display the Gross Pay

Step 2) Input: You should know how the Input looks like: (5 Points)

Enter Employee's Full Name: Ogar Haji
Enter Hours Worked: 40
Enter Hourly Rate: 10

Step 3) Processing and Calculation: The program will process each record and Calculate Gross Pay: (5 Points)

Gross Pay = Hours * Rate

Step 4) Output: You should know how output should look like: (5 Points)

********* Calculate Gross Pay Project *********

The Employee Ogar Haji Gross Pay is = $ 400

********** End of Program **********

Step 5) Flowchart: Draw a Flowchart for Gross Pay program.(5 Points)

Start

1) Declare the Variables to be used in the project

2) print to output "Title/Heading"

3) Prompt user "Enter Employee's Full Name: "

4) Read the fullName and store it in a variable

5) Prompt the user "Enter Hours Worked: "

6) Read hours worked and store it in hours variable

8) Prompt the user "Enter Hourly Rate: "

9) Read Hourly Rate and store it in a rate variable

11) Calculate Gross Pay = hours * rate

12) print to output "Your Gross Pay is = $", grossPay

13) print to output "******* End of Program *******"

End

Step 6) PseudoCode: print a PseudoCode for the Program.(5 Points)

1) Declare the variables to be used in the program
2) print to Console "The Title or Heading of the program "
3) Prompt the user "Enter Employee's Full Name: "
4) Read from Console fullName and store the String in a variable
5) Prompt the user "Enter Hours Worked: "
6) Read from console hours worked and store in a String variable
7) Convert String hours variable to integer value
8) Prompt the user "Enter Hourly Rate: "
9) Read from console the rate and store in a String variable
10) Convert String rate variable to double value
11) Calculate Gross Pay = Hours * Rate
12) print to Console "The Employee Gross Pay is = $", gross_pay
13) print to Console "******* End of Program *******"

Step 7) Code the Program in Java by referencing the Flowchart or Pseudocode you designed above and Save it as CalculateGrossPay to USB.

a) Type the following Java code in Eclipse IDE.

Code for the first part of "CalculateGrossPay" project follows:

```
package calculategrosspay;

// import the Java Scanner class needed to Read and Write
import java.util.Scanner;

/*
 ************************************************************
   Purpose of the Project:
   a) This Interactive Project will prompt the user to enter
      his/her (First Name, Last Name, and Hours Worked and Rate)
      then it will read the text entered and store it
      in its variables. Then it Calculates the Gross Pay.
   b) Project Name: CalculateGrossPay
   c) Date: Saturday, December 28, 2016
   d) Programmer: Instructor – Ogar Haji
 ************************************************************
```

```
*/
public class CalculateGrossPay {

public static void main(String[] args) {

    // 1) Declare the variables to be used in the project
    String firstName, lastName;
    int hours = 0;
    double rate = 0.0;
    double grossPay = 0.0;

    // 2) Instantiate an object from the Scanner class
    Scanner input = new Scanner(System.in);

    // 3) Prompt the User to Enter his/her First Name
    System.out.print("Enter your First Name:\007 ");

    // 4) Read First Name from keyboard and store it in a variable
    firstName = input.nextLine();

    // 5) print First Name to the output
    System.out.println("Your First Name is: " + firstName);

    // 6) Prompt the User to Enter his/her Last Name
    System.out.print("Enter your Last Name: \007");

    // 7) Read his/her Last Name from keyboard and store it in a variable
    lastName = input.nextLine();

    // 8) print Last Name to the output
    System.out.println("Your Last Name is: " + lastName);

    // 9) Prompt the User to Enter Hours Worked
    System.out.print("Enter Number of Hours Worked: \007");

    // 10) Read Hours Worked from keyboard and Store it in a variable
    hours = input.nextInt();

    // 11) print Hours Worked to the output
    System.out.println("Hours Worked: " + hours);
```

```
        // 12) Prompt the User to Enter Hourly Rate
        System.out.print("Enter Hourly Rate: ");

        // 13) Read Hourly Rate and Store it in a variable
        rate = input.nextDouble();
        // 14) print Hourly Rate to the output
        System.out.println("Hourly Rate: " + rate);

        // 15) Calculate Gross Pay
        grossPay = hours * rate;

        // 16) print grossPay to the output screen
        System.out.printf ("Gross Pay is = $%,.2f %n" , grossPay);
    }
}
```

Step 8) Click **Run Project**  button to Start Running the program

The following output appears on the Left side of the screen with the Input you entered and the correct calculated GrossPay $400.

If any Syntax Errors Found Do Next Step 9:

```
Your First Name is: Ogar
Enter your Last Name: Haji
Your Last Name is: Haji
Enter Number of Hours Worked: 40
Hours Worked: 40
Enter Hourly Rate: 10.50
Hourly Rate: 10.5
Gross Pay is = 420
```

Step 9) Debug the Program: Debug or Correct any Syntax Errors until you have a clean Compiled program. (5 Points) (Clean compiled program means No Errors in the program).

Step 10) Test the Program: Test the Program with Test Data. (5 Points)

Repeat Step 10) Test the program many Times and Test the Program again and again until All conditions are tested:

```
Your First Name is: Mary
Enter your Last Name: Smith
Your Last Name is: Smith
Enter Number of Hours Worked: 30
Hours Worked: 30
Enter Hourly Rate: 11.23
Hourly Rate: 11.23
Gross Pay is = 336.9
Mary Smith 30 11.23 336.9
Your Gross Pay is = $336.90
```

Step 11) Documentation (5 Points): You have to add more comments to the Program (like Comments about the Purpose of the Program, Your Name and the Date the Program was written.)

```
##################################################################
# Purpose of the Program:
#   a) This Program will calculate Gross Pay.
#   b) It will ask the User to Enter Employee's Full Name:
#   c) It will ask the User to Enter Number of Hours Worked
#   d) It will ask the User to Enter Hourly Rate.
#   e) The program will calculate Gross Pay.
#       Gross Pay = Hours * Rate
#   f) Display Gross Pay
##################################################################
```

Step 12) Print a Copy of Java Code along with screen printout of the Running program. Submit to your Instructor the Print Copy and the screen Printout (Snaps) along with the following: (Which you did on Paper)

Copy the Java Code and the result of the program and Paste it in Microsoft Word program:

1) Purpose of the Program.
2) Input: how the Input looks like

3) Processing and Calculations
4) Output: how the Output will look like
5) Flowchart 6) Pseudocode
7) Java Code and
8) Print out copy of Java code and Output after running the program.
Submit the Programs on Time.
Remember Points will be deducted (20%) for Programs submitted Late.

Important Note:
1) Do Steps 1 thru 7 on Paper.
2) Then Get into NetBeans IDE Code Editor
3) Type the Java code.
4) Save All the Files
5) Run the Program and Test it with Test Data for All Conditions.

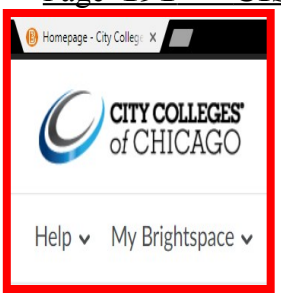
Modify the Project and add the following to print to the Console using Format Specifiers (%s, %d, %,.2f, %n) with printf() method.

To print the Results using Format Specifiers: %s %d %f %.2f %n

```
// 19) Using Format Specifiers (%s, %d, %f, %n) to print out the output
System.out.printf ("%s      %s     %d     %f    $%,.2f ",
                    firstName, lastName, hours, rate, grossPay);
```

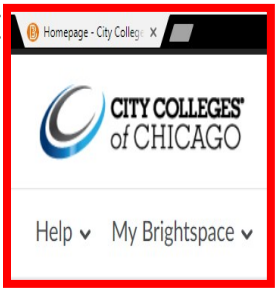
```
// 20) print the Formatted Gross Pay to output screen
System.out.printf("Gross Pay is = $%,.2f %n" , grossPay);
```

When Modifying a Project
Do Only 1 Modification at a Time



Upload Your Weekly Assignments: To Brightspace Correct Weekly Assignments Folder

+++ Review +++



Lesson 72 : How to Upload Your Weekly Assignments to Brightspace Correct Weekly Assignments Folder?

You have to Upload your Weekly Assignments and Homework to Brightspace correct Weekly Assignments Folder as following:

1) Copy the Java Code from NetBeans to Word document:

1. Copy the **Java Code from** the **NetBeans IDE** and **Paste it into** the **Microsoft Word Document.**
2. In NetBeans IDE, press **Ctrl+A (select All)** to **select All** the **Java code**.
3. Press **Ctrl+C (Copy)** to **Copy** the **selected Java code** into **computer memory RAM.**

2) Paste the Java Code into Microsoft Word:

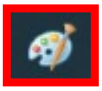
1. Get into **Microsoft Word document then** press **Ctrl+V (paste) to Paste** the **copied Java code** from **memory** into **Word document.**
2. Press **Ctrl+Home (go to the Top of Document)** and **type** your **Full Name** at the **top of document followed by** the **Java File Name** in **size 20 and bold.**

3) Print the Screen of the Output of Java NetBeans:

1. Run the **Java project** and **make sure the program** is **running** with **correct output.**
2. Press **PrintScreen button** 

to **capture** the **output screen shot.**

4) Paste the Print Screen of Java output into Paint program:



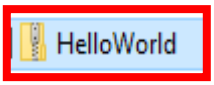
1. Get into **Paint program** and press **Ctrl+V (paste) to Paste** the **screen shot** in **Paint** program.
2. Inside the **Paint program,** Click **Select** 

icon and then **Select only** the **Output** of the **Java project.**
3. Press **Ctrl+C (Copy)** to **Copy** the **selected output image.**

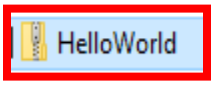
5) Get back into Microsoft Word program:

1. Go back to **Word Document,** press **Ctrl+End (End of Document)** to **go to End of document.**
2. In the **Word document,** press **Ctrl+V (Paste)** to **paste** the **Java output** there.
3. Save the **Word Document** as the **Name of the Java project** and in this **example (Save File as HelloWorld project)**

6) To Compress or Zip the Java project:

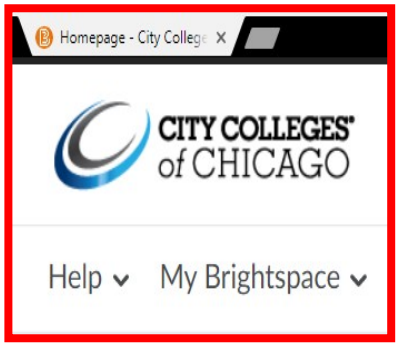


1. Right-Click on the **Java project (HelloWorld)** that is **saved on your computer**.
2. Point to **Send to,** then **click on Compressed (Zipped) Folder** and you will have **another File** which is Compressed or Zipped.



7) To Upload the 2 Files to Brightspace:

1. Log on Brightspace with your **User Name** and **Password**
2. Click on your course **CIS 144 Java** to **Select it**.
3. Click on **Assignments**
4. Click on **Assignments**
5. Click on **Week 03 Assignments Folder**
6. Click on **"Add a File"** button
7. Click on **"My Computer"**
8. Click on **Upload** button



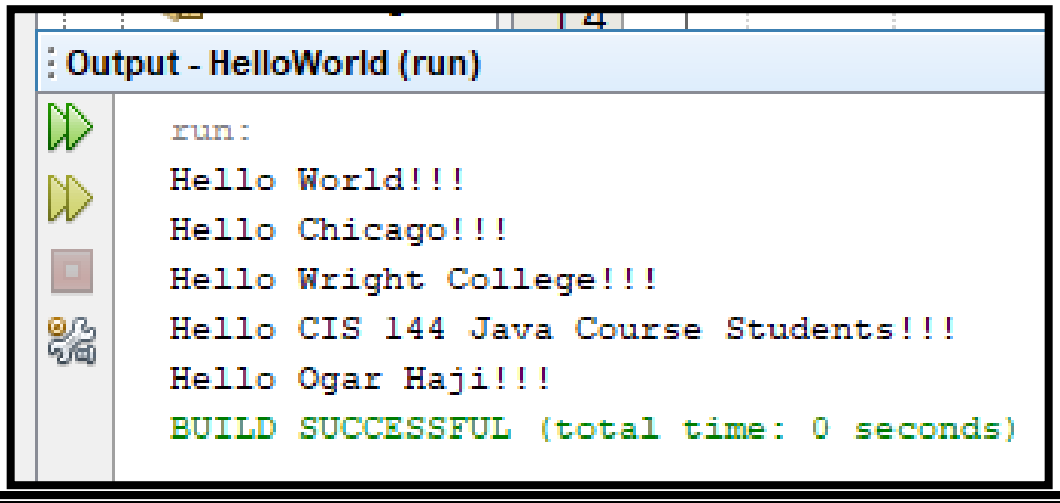
9. Go to the **location** where you **saved the Java project "HelloWorld".**
10. Click on the **File** or **Folder (HelloWorld)**
11. Click on **Add** button and the **File** or **Folder** will be **added** to the **Week 01 Assignments Folder.**

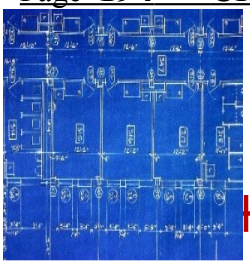
Note: Always **Upload** to **Brightspace 2 Files** of **same Java Project:**
1) The **Microsoft Word Document** of the **Java Project Code along with the Java Output Screen shots.**
2) The **Compressed** or **Zipped File** or **Folder** of **Java Project**.

The Java project "HelloWorld" code in Word Document along with the Output Screen Shots appear as following:

Ogar Haji (Your Full Name)
CIS 144 Java
Project Name: HelloWorld

```
/*
Project Name: HelloWorld
This Java project will print the message "Hello World" to screen
Programmer: Instructor + Ogar Haji (Type your Full Name)
Date:   June 01, 2017
*/

package helloworld;

public class HelloWorld {
    public static void main(String[] args) {
        // This project will print the Literal String "Hello World" to output screen

        System.out.println ("Hello World!!!");
        System.out.println ("Hello Chicago!!!");
        System.out.println ("Hello Wright College!!!");
        System.out.println ("Hello CIS 144 Java Course Students!!!");
        System.out.println ("Hello Ogar Haji!!!");
    }
}
```





Java is Object-Oriented Programming (OOP) Language

++(Read and Study This Lesson)++

Lesson 73 – What is Object-Oriented Programming Language or Java OOP?

Object means **a real-world entity** such as a **House**, a **Car**, a **TV**, a **Person** and so on.

Object-Oriented Programming (OOP) is a methodology to design a program **using classes** and **objects**. It simplifies development and maintenance by providing some concepts:

1. **Class:** A **class** is the **Blueprint** or **Template** from which **individual Objects** or **instances** are **created**.
2. **Object:** An **Object** is an **Instance** of a **class.** You can **create objects** from **Classes** like **creating** the **Object 'input'** from **Class Scanner**.

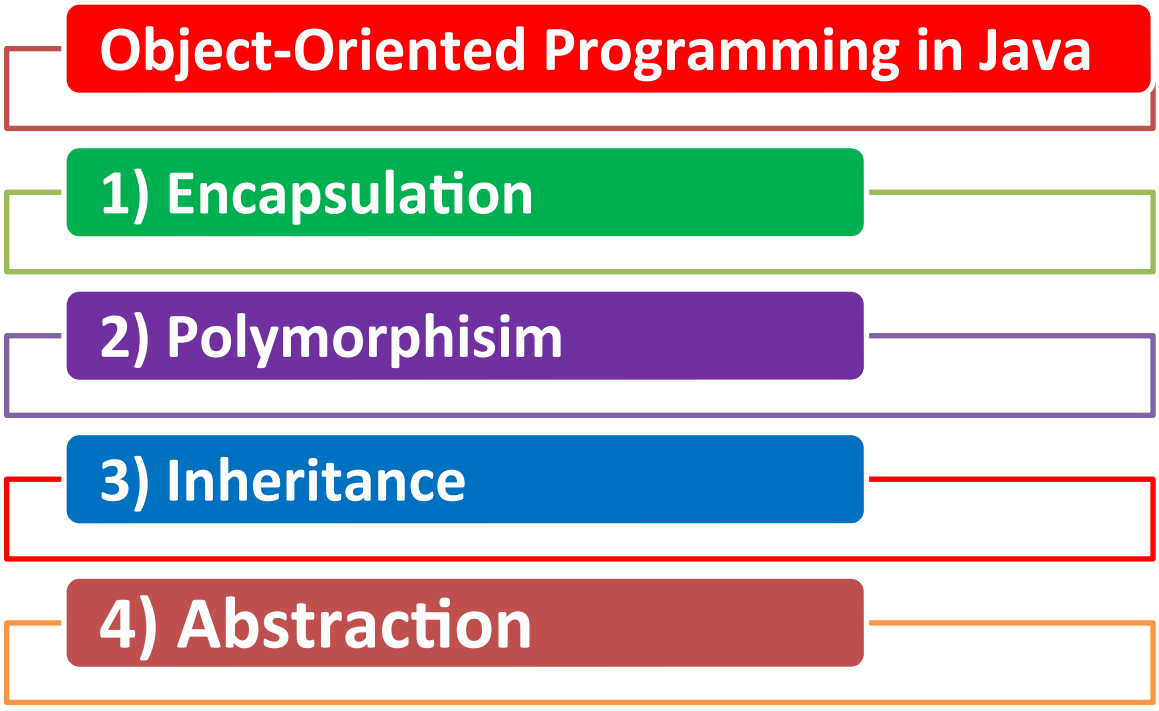
```
Scanner input = new Scanner (System.in);
```

3. **Encapsulation:** **Encapsulation** is **Hiding code** and **data together** into a **single unit**.
4. **Polymorphism:** **Polymorphism** is that you can **create many methods** with **same name** but **different parameters** or **signature**.

```
public static void calculateSum(int number1, int number2) {
        int sum = number1 + number2;
}
public static void calculateSum(double number1, double number2) {
        double sum = number1 + number2;
}
```

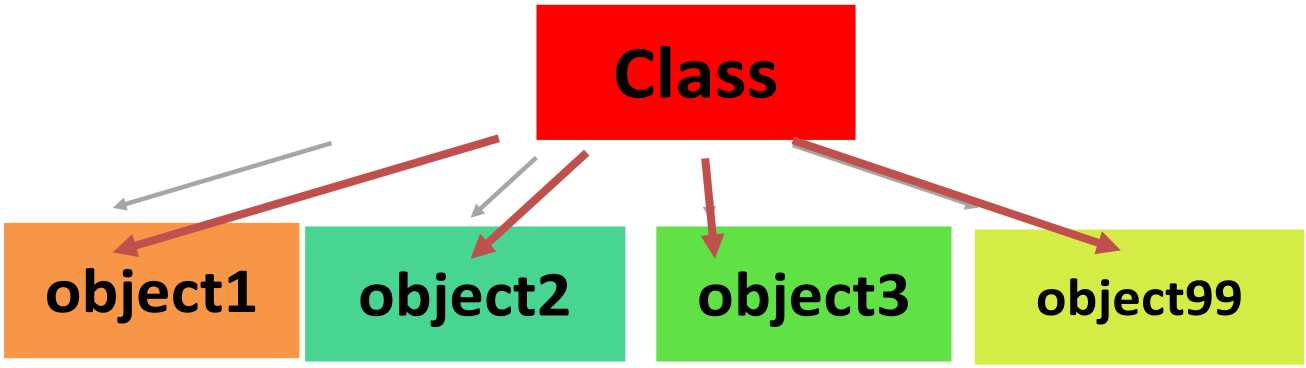
5. **Inheritance:** **Inheritance** is when an **object extends** or **inherits** or **acquires all** the **properties** and **behaviors** of a **parent object**.

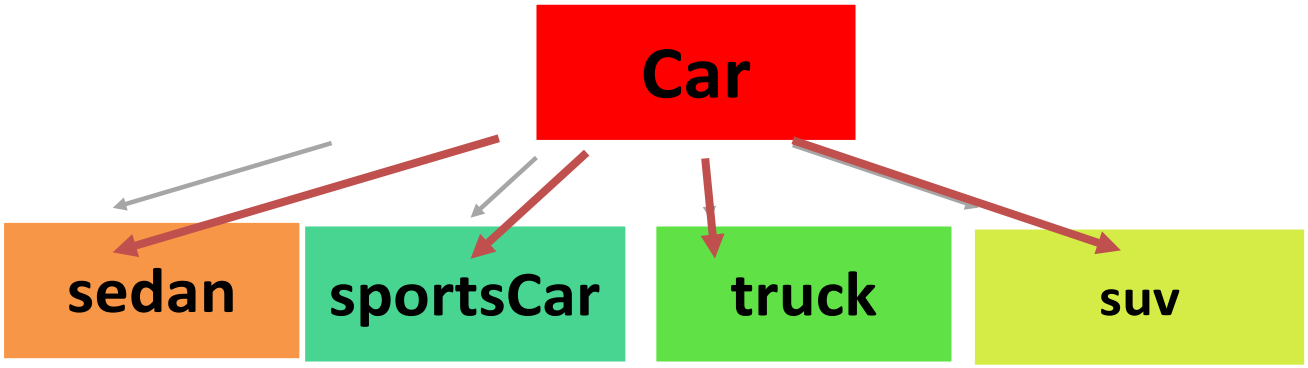
6. Abstraction: Hiding internal details and showing **functionality**.



You have to create a Class (like Car) first and then you can create as many Objects (chevy, ford, buick) from the class using 'new' keyword.

```
public class Car {
// In the main( ) method, you can Create Objects from the
// Class Car.
Car sportsCar = new Car();
```



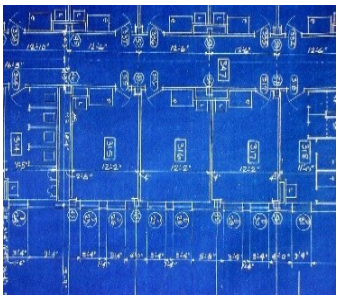


Java is an **Object-Oriented Programming Language**.

Objects contain **properties** or **data** and **methods( )**.

Properties or **Instances** or **Fields** are like **variables**.

Methods( ) are **actions** that the **object can perform**.



by Unknown Author is licensed under

A **Class** is the **Blueprint** or **Template** which you can use **to create** many **Objects**. For example, you will **create** the **blueprint of a car** and out of this blueprint you can **create many cars (objects).**

You have to **create** the **Class first** and then you **create** an **Object** of the class. Just like we **created** an **Object** '**input**' from the **class Scanner**.

```
Scanner input = new Scanner (System.in);
```

An **Object** is an **Instance** of a **class**.

Objects have **States** and **Actions**.

States are the **variables**, **fields,** or **attributes** of the **objects**.

Actions are the **methods( )** of the **objects**.

We call **methods( )** on **objects**.

To Create a Class Called Car

Lesson 74 – How to Create a Class called 'Car' using Object-Oriented Programming or Java OOP?

We can **create** a **class** called **Car** and an **object** called **myCar**.
The **Attributes** or **Fields** of the **Car** class can be: **make**, **model, color**, **price**.
The **Methods()** or **actions** of the **object house** can be: **drive()**, **stop(), speed(), fillWithGas()**.

To Create a Class called Car:

```
public Class Car {
```

After Creating a **class** called **Car,** then **create** an **object** called **myCar**.
To Create an **object** called '**myCar**' from **class Car** in main()

```
Car  myCar  =  new Car();
```

The **Attributes** or **Fields** of the **Car** class can be: **make**, **model**, **color**, **price**.
The **Methods()** or **actions** of the **object car** can be: **drive()**, **stop(), fillWithGas(), speed()**.


```
// To Create a class called Car with some of private instances or fields
// Make the fields private so they will be accessed only within the class
public class Car {
    // 1) Declare private instance fields or attributes
    private String make;
    private String model;
    private int year;

  // 2) Define and code the drive() method
public static void drive() {
        System.out.println("The car is moving forward");
```

}

by Unknown Author licensed under

To Create a Class Called Student

by Unknown Author licensed under

Lesson 75 – How to Create a Class called 'Student' using Object-Oriented Programming or Java OOP?

We can **create** a **class** called **Student** and an **object** called **student1**.

The **Attributes** or **Fields** of the **Student** class can be: **firstName**, **lastName**, **age**, **phoneNo**.

The **Methods()** or **actions** of the **object student1** can be: **study()**, **goToSchool(), goToWork().**

To Create a Class called Student:

```
public Class Student {
```

After Creating a **class** called **Student,** then **create** an **object** called **student1**.

To Create an **object** called '**student1**' from **class Student** in main()

```
Student  student1  =  new Student ();
```

// To **Create** a **class** called **Student** with some **private instances or fields**.

// Make the **fields private** so they will be **accessed only within the class**

```
public class Student {
    // 1) Declare private instance fields or attributes of class Student
    private String firstName;
    private String lastName;
    private int age;
    private String phoneNo;
   // 2) Define and code the study() method
public static void study() {
```

```
    System.out.println("The Student is studying a book about Java");
}
```

To Create a Class Called House

Lesson 76 – How to Create a Class called 'House' using Object-Oriented Programming or Java OOP?

// To **Create** a **class** called **House** with some **private instances** or **fields**
// Make the **fields private** so they will be **accessed only within the class**
The **Attributes** or **Fields** of the **House** class can be: **numberOfRooms**, **numberOfBathRooms**, **address**, **price**.
The **Methods()** or **actions** of the **object myHouse** can be: **paint()**, **builtGarage(), mowingTheGrass().**

To Create a Class called House:

```
public Class House {
```

After Creating a **class** called **House,** then **create** an **object** called **myHouse**
To Create an **object** called '**myHouse**' from **class House** in main()

```
    House myHouse  =  new House ();
```

```
public class House {
   // 1) Declare private instance fields or attributes of class House
   private int numberOfRooms;
   private int numberOfBathRooms;
   private double price;
   private String address;
```

```
  // 2) Define and code the paint() method
public static void paint() {
    System.out.println("I will paint the house Green soon");
}
```

```
  // 2) Define and code the builtGarage () method
public static void builtGarage() {
```

```
System.out.println ("We will build a Garage next Month");
}
```

public
private
protected
default

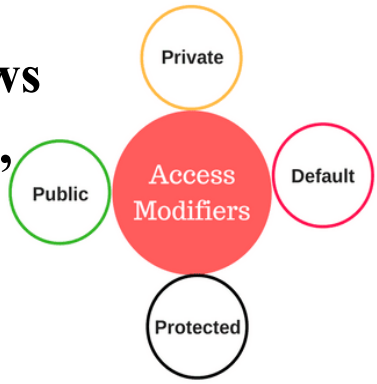
Java Access Modifiers are: public, private, protected and default

++(Read and Study This Lesson)+

public
private
protected
default

Lesson 77 – What are the Access Modifiers in Java?

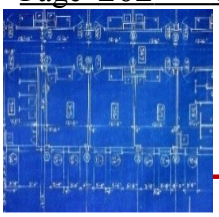
The 4 Access Modifiers in Java Language allows or prevents the access to Classes, Constructors, Data Members and Methods in another class.



The 4 Access Modifiers in Java language are:

1) **public: public** access modifier **allows access** to **all classes, Data Members,** and **methods** in the **classes by all other classes** in the **project**.
2) **private: private** access modifier **restrict access** to that **class only So,** the **outside members cannot access** the **private members**.
3) **protected: protected** access modifier **allows access within** the **same package classes** and also **from** other **sub classes of other packages**.
4) **default: If you do not declare a class or variables explicitly** then they will be **declared as default. Default**

classes and **methods are visible** to the **classes with** the **same package**.



UML (Unified Modeling Language) Java (OOP) Programming Language ++(Read and Study This Lesson) +

Lesson 78 – What is UML (Unified Modeling Language) in Java?

UML stands for Unified Modeling Language and is used for modeling Classes as diagrams.

UML diagram is used to **visualize** a **Class Attributes (Variables** or **Fields)** and **Methods (Actions or Operations)** of a **class.**

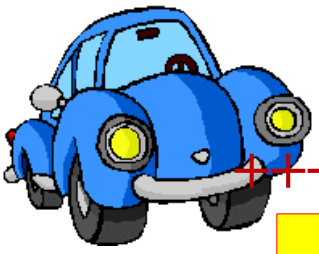
Top Compartment (Class Name)
Middle Compartment (Attributes or Fields)
Bottom Compartment (Methods or Actions)

Applying **UML Diagram** for the **Class BankAccount**

BankAccount
- firstName : String (- means private) - lastName : String - balance : double
+ setFirstName (firstName : String) + setLastName (lastName : String) + setBalance (balance : double) + getFirstName : String (+ means public) + getLastName : String + getBalance : double

- sign in front of **Attributes** or **Fields** means these **Fields** are **private**.

+ sign in front of **Methods** means these **methods** are **public**.



Creating Car Class + Java Object Oriented Project +++(Do Lab Exercise 1) 100 Points+++

by Unknown Author is licensed under

Do Lab Exercise 1

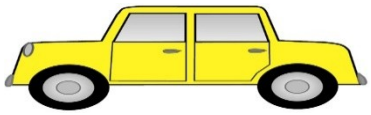
Lesson 79 - How to Create a Java Class called Car Class using Java OOP?

Problem or Project: Design and Code in Java Language the project to Use Object-Oriented Programming to create a Class called Car and implement Car Properties and Methods.

Do Lab Exercise

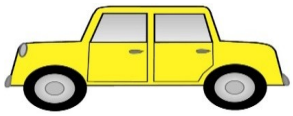
Start Page | Output - CarTest (run) | CarTest.java | Car.java

```
run:

************************************
******** Car Class project ************

************************************
Please, Enter car Make: Toyota
Please, Enter car Model: Camry
Please, Enter car Year: 2016

The following are information about the car:Toyota Camry 2016
Press 'Q' to Quit or any other key to continue: y
Please, Enter car Make: Chevy
Please, Enter car Model: Impala
Please, Enter car Year: 2019

The following are information about the car:Chevy Impala 2019
Press 'Q' to Quit or any other key to continue: q

************************************
******* End of Job : Your Name **********

************************************
BUILD SUCCESSFUL (total time: 1 minute 2 seconds)
```



by Unknown Author is licensed under

A) To Create the class Car, Do the following:



by Unknown Author is licensed under

1) **Create** a **Class** called **Car:**

```
public class Car {
```

2) Declare as private the instance variables or fields of class **Car**

```
private String make;
private String model,
private int year;
```

3) Create a Car constructor passing 3 parameters.

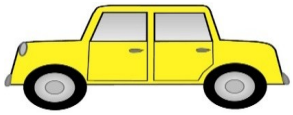
```
public Car (String make, String model, int year) {
    this.make = make;          // use this keyword for current object
    this.model = model;
   this.year = year;
}
```

4) Define and code **setMake()** method in the object.

```
public void setMake (String make) {
    // Store the parameter in make field using this keyword
       this.make = make;
}
```

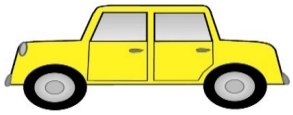
5) Define and code **setModel()** method in the object.

```
public void setModel (String model) {
    // Store the parameter in make field using this keyword
       this.model = model;
}
```



6) Define and code **setYear()** method in the object.

```
public void setYear (int year) {
    // Store the parameter in year field using this keyword
       this.year = year;
}
```



7) Define and code **getMake()** method in the object.

```
public String getMake() {
     // Return the String value of car Make to caller
        return make;
}
```

4) Create an object '**myCar**' from class Car in main() method

```
Car  myCar = new Car();
```

5) Print out the initial value of first name which is '**null**' by calling the **getModel()** method

```
System.out.printf ("Initial value of car Model is : %s %n",
```

myCar.getModel());

B) To Create the class CarTest (Which is also the Driver), Do the following:

Follow the following steps:

1) Get into **NetBeans IDE**
2) **Create CarTest main()** program which will **contain** the **main()** method**:**
3) **Create** the class **Car** which will **contain** the **objects fields** in it and also the **object methods:**
   - **a. RC** the **package** or **folder**
   - **b. Point** to **new**
   - **c. C Java Class**
   - **d. Type Car**
   - **e. C Finish**

Do Lab Exercise

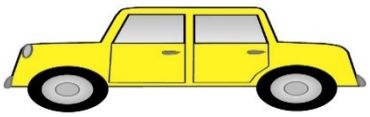
4) NetBeans IDE will **create 2 Tabs: CarTest** and **Car**
5) Click on **Car tab** 

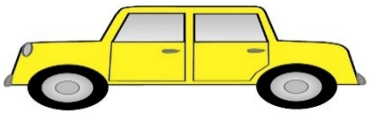
, type the following **Java code:**

1) Type the following Java program and
2) Save file as CarTest

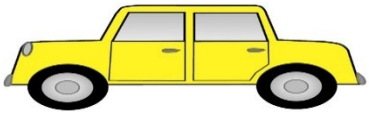
Do Lab Exercise

```
package cartest;
// Import the Class Scanner
import java.util.Scanner;
// This is the Car Class Method: Car
public class Car {
    // 1) Declare private instance fields or attributes
    private String make;
    private String model;
    private int year;
    // 2) Create an object 'input' from the class Scanner
    Scanner input = new Scanner (System.in);

    // 3) Create a Car constructor passing 3 parameters
```


by Unknown Author is licensed under

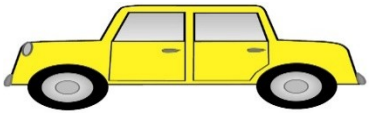

by Unknown Author is licensed

```
public Car (String make, String model, int year) {
    this.model = model;          // use this keyword for current object
    this.make = make;
    this.year = year;
}

// 4) Define and code the getMake() method

public String getMake() {
    return this.make;
 }
// 5) Define and code the getModel() method
public String getModel() {
    return this.model;
 }
```



by Unknown Author is licensed under

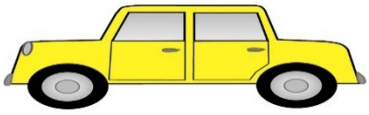
```
// 6) Define and code the getYear() method
public int getYear () {
    return this.year;
}
// 7) Define and code the setModel() method
public void setModel(String model) {
   this.model = model;
}

// 8) Define and code the setMake() method
public void setMake(String make) {
    this.make = make;
}
```



by Unknown Author is licensed under

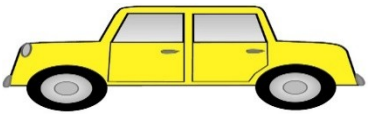
```
// 9) Define and code the setYear() method
public void setYear(int year) {
  // Check for invalid year < 2000 and > 2019
  while (year < 2000 ||  year >2019) {
    System.out.print("Please enter valid year (Between 2000 and 2019: ");
```

```
      year = input.nextInt();
   }

   this.year = year;  // Assign the valid year
}
```

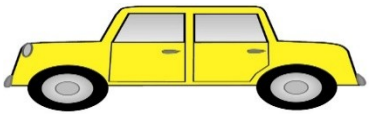


by Unknown Author is licensed under

```
// 10) Use the toString() method to print formatted string
  public String toString(){
     System.out.println();
     return this.make + " " + this.model + " " + Integer.toString(this.year);
  }
}
// This is the main Method: CarTest
public class CarTest {
   public static void main(String[] args) {

      // 1) Call print Headings method
      printHeadings();

      // 2) Call create the Car Class Objects method
      createCarClass();

      // 3) Print footings method
      printFootings();

   }  // End of main method

   // 1) Define and Code the printHeadings method
   public static void printHeadings() {
      System.out.println("\n***********************************");
      System.out.println("******** Car Class project ***********");
      System.out.println("\n***********************************");
   }
```

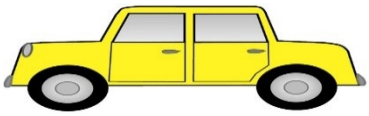


by Unknown Author is licensed under



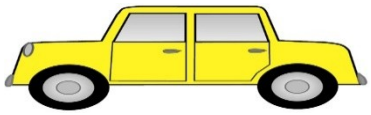
by Unknown Author is licensed under

```
// 2) Define and Code the create Car class method
public static void createCarClass() {
    // 1) Declare the variables to be used in the project
    boolean quit = false;
    String model, make;
    int year;

    String toQuit;

    //2) create an object 'input' from the Class Scanner
    Scanner input = new Scanner (System.in);

    // 3) Create a Constructor of empty objects
    Car myCar = new Car ("", "", 0);
```



by Unknown Author is licensed under

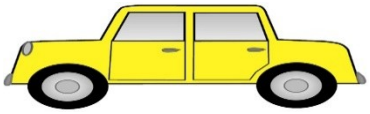
```
    // 4) Use while statement to loop until the user press 'Q' to Quit
    while (!quit) {
        // 5) Prompt the user for data about Car and Read them
        System.out.print ("Please, Enter car Make: ");
        make = input.nextLine();

        System.out.print ("Please, Enter car Model: ");
        model = input.nextLine();

        System.out.print ("Please, Enter car Year: ");
        year = input.nextInt();

        // 6) Call set methods and passing the parameters to Assign Data to objects
        myCar.setMake(make);
        myCar.setModel(model);
        myCar.setYear(year);
```



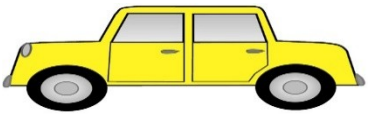
by Unknown Author is licensed under

```
        input.nextLine();  // Read the empty return key

    // 7) Print the Information about the car entered by user
    System.out.println("The following are information about the car:"+myCar);
        System.out.print ("Press 'Q' to Quit or any other key to continue: ");
```

```
            toQuit = input.nextLine();

            // 8) Check if the user entered 'Q' to quit
            if (toQuit.equalsIgnoreCase("Q")) {
                quit = true;
            }
        }
    }
    // Define and Code the printFooting method
        public static void printFootings(){
        System.out.println("\n*************************************");
        System.out.println("***** End of Job : Ogar Haji (Your Name) *****");
        System.out.println("\n*************************************");
    }
}
```

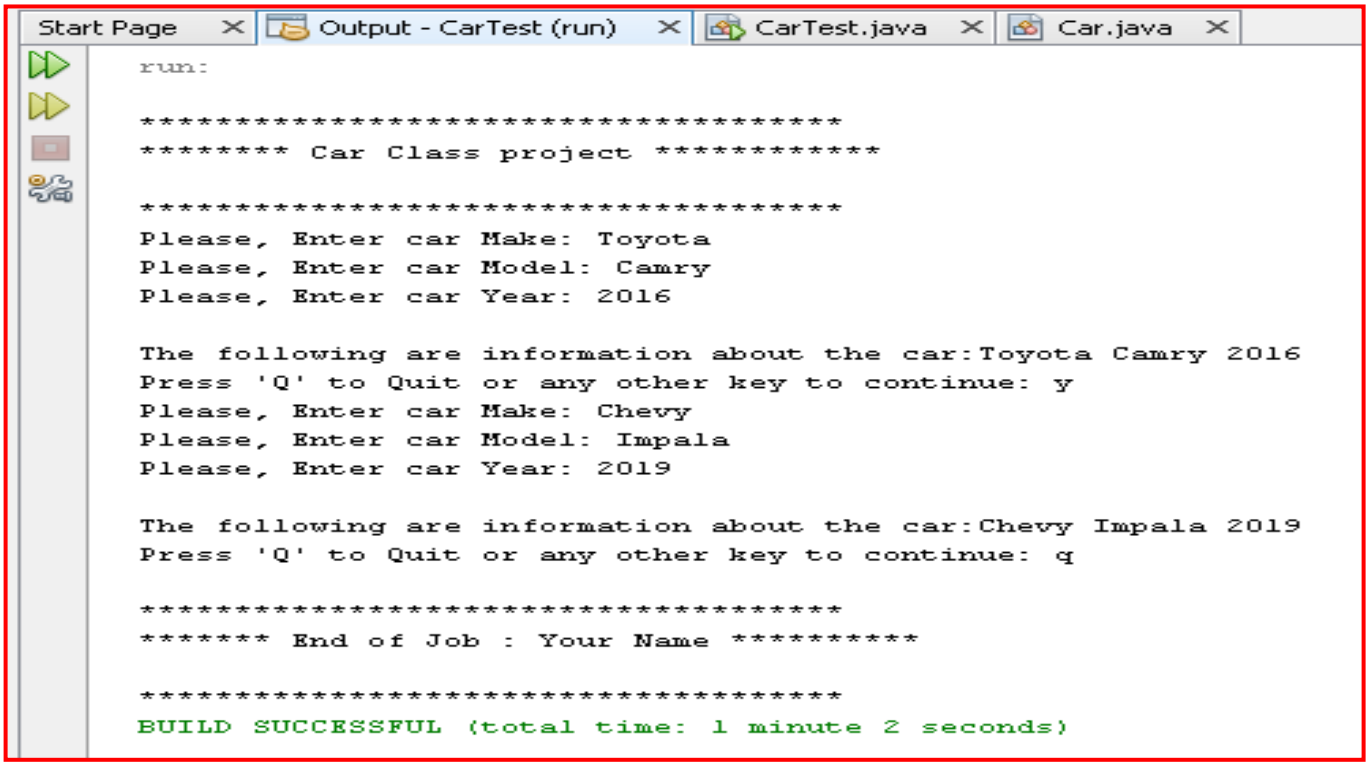


by Unknown Author is licensed under



by Unknown Author is licensed under

The output this Java project will look like the following:



1) **Modify the Car project to add Color property to the Car private fields.**
2) **Modify the Car project to add Price property to the Car private fields.**
3) **Modify the Car Class Java project to accept only the following cars Make (Chevy, Ford and Toyota)**

Java Object-Oriented Programming
Create Bank Account Class
+++(Do Lab Exercise 2) 100 Points+++

Do Lab Exercise 2

Lesson 80+ How to Create a Java Class called BankAccount using Java OOP?

Problem or Project: Design and Code in Java Language a project to use Object-Oriented Programming to create a Class called 'BankAccounct' for Customers and implement Car Properties and Methods.

To Create a **class** called **BankAccount** for **customers** in a **bank,** we will have only **1 field** called **First Name** to make it simple then we will add **new fields** as we go along.

A) To Create the class BankAccount, Do the following:

1) Create a **Class** called **BankAccount:**

```
public class BankAccount {
```

2) Declare as private the instance variables or fields of class **BankAccount**

```
private String firstName;
```

3) Define and code **setFirstName()** method in the object

```
public void setFirstName (String firstName) {
    // Store the parameter in firstName field using this keyword
      this.firstName = firstName;
}
```

4) Define and code **getFirstName()** method in the object

```
public String getFirstName() {
    // Return the String value of first name to caller
      return firstName;
}
```

5) Create an object '**bankAccount1**' from class BankAccount in main()

```
BankAccount  bankAccount1 = new BankAccount();
```

6) Print out the initial value of first name which is '**null**' by calling the **getFirstName()** method

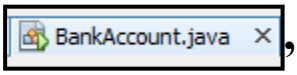
```
System.out.printf("Initial value of First Name is : %s %n",
                  bankAccount1.getFirstName());
```

B) To Create the class BankAccountTest (Which is also the Driver), Do the following:

Follow the following steps:

1) Get into **NetBeans IDE**
2) **Create BankAccountTest main()** program which will **contain** the **main()** method**:**
3) **Create** the class **BankAccount** which will **contain** the **objects fields** in it and also the **object methods:**
   - **a. RC** the **package** or **folder**
   - **b. Point** to **new**
   - **c. C Java Class**
   - **d. Type BankAccount**
   - **e. C Finish**

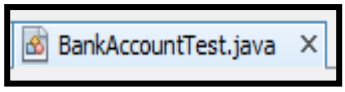
Do Lab Exercise

4) NetBeans IDE will **create 2 Tabs: BankAccountTest** & **BankAccount**
5) Click on **BankAccount tab** 

, **type** the following **Java code:**

```
package bankaccounttest;

public class BankAccount {
// 1) Declare as private the instance variables or fields of class BankAccount
   private String firstName;

   // 2) Define and code getFirstName() method in the object
   public String getFirstName() {
      // 3) Return the String value of first name to caller
      return firstName;
   }

   // 4) Define and code setFirstName() method in the object
   public void setFirstName (String firstName){
       // 5) Store the parameter in firstName field
```

```
        this.firstName = firstName;
    }
}
```

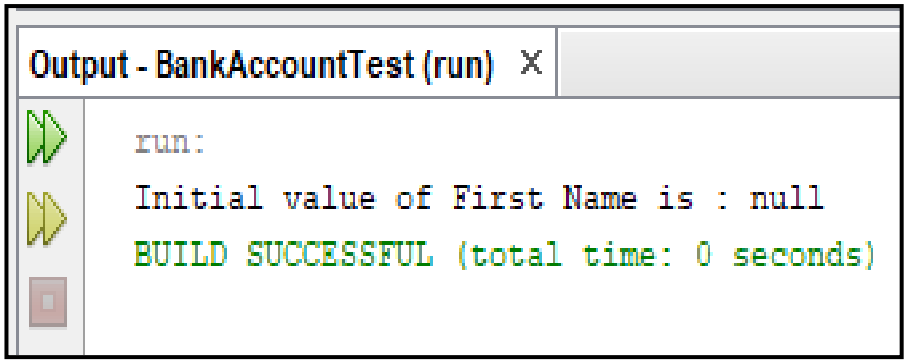
10)Click on **BankAccountTest tab**  and **type** the following **Java code:**

Do Lab Exercise

```
package bankaccounttest;
// import the Scanner class from java.util package
import java.util.Scanner;
public class BankAccountTest {
    public static void main(String[] args) {
        // 1) Create an object 'input' from class Scanner using new
        Scanner input = new Scanner (System.in);

        // 2) Create an object 'bankAccount1' from class BankAccount
        BankAccount  bankAccount1 = new BankAccount();

        // 3) Print out the initial value of first name which is 'null'
        // by calling the getFirstName() method
        System.out.printf ("Initial value of First Name is : %s %n",
                           bankAccount1.getFirstName());
    }

}
```

Run the Java program and the output appears below:

When a String Field or String variable is Not assigned a value, Java will assign to it by default the "null" value as shown below.



Modify the BankAccount Java project by doing the following:

1) Add another **private String field** called **"lastName"** and **declare** it as **private**.

```
private String lastName;
```

2) Define and code **getLastName()** method in the object

```
public String getLastName() {
    // 3) Return the String value of last name to caller
    return lastName;
}
```

3) Define and code **setLastName()** method in the object

```
public void setLastName (String lastName) {
    // 4) Store the parameter in lastName field
    this.lastName = lastName;
}
```

4) Prompt the **user** to **Enter the Last Name**

```
// 5) Prompt the User to Enter Last Name
System.out.print("Please, Enter the user Last Name: ");

// 6) Read and store the user Last Name
String userLastName = input.next();
```

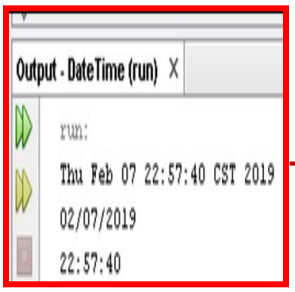
// 7) Call **bankAccount1 setLastName()** method to store last name in it

bankAccount1.setLastName (userLastName);

// 8) **Print** out the **Last name stored** in the **object bankAccount1**

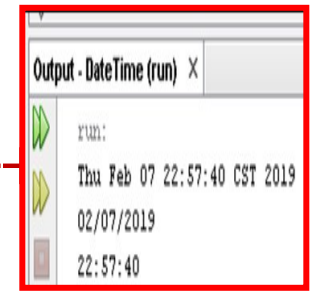
System.out.printf("First Name of the object bank account is: %s %n ",

bankAccount1.getLastName());



Insert Computer Date and Time In Java Project

++(Do Lab Exercise 3)100 Points+-



Do Lab Exercise 3

Lesson 81 Ex : How to Insert Computer Date and Time using Java?

Problem or Project: Design and Code in Java Language the project to Display Current Date and Time to the screen.

// **1)**To Insert **Computer Date and Computer Time**, import the following classes:

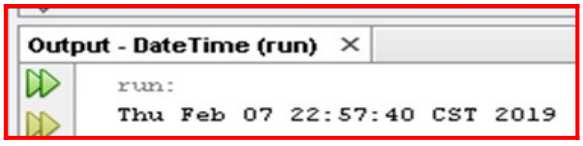
import java.util.*;
import java.text.SimpleDateFormat;

// **2)** Create an instance object called **dateFormat** from the class **SimpleDateFormat**

SimpleDateFormat dateFormat = new
SimpleDateFormat("yyyy/MM/dd");

// **3)** Create an instance object called **'date'** from the **class Date**

Date date = new Date();



// **4)** Print out the date in the dateFormat Format

System.out.println(dateFormat.format(date)); // prints 2017/10/07

// **5)** Create an instance object called **timeFormat** from the class **SimpleDateFormat**

SimpleDateFormat timeFormat = new

```
                                SimpleDateFormat("HH:mm:ss");
Date timeNow = new Date();
```

// 6) Print out the timeFormat

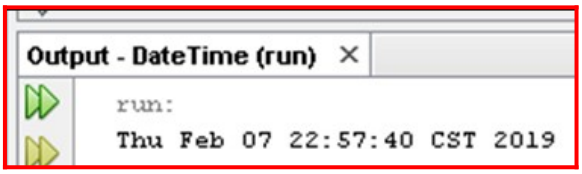
```
System.out.println(timeFormat.format(timeNow)); // prints 16:22:39

SimpleDateFormat dateFormat2 = new
                                SimpleDateFormat("MM/dd/yyyy");
Date date2 = new Date();
System.out.println(dateFormat2.format(date2)); // prints 10/21/2017
```

1) Type the following project in NetBeans and

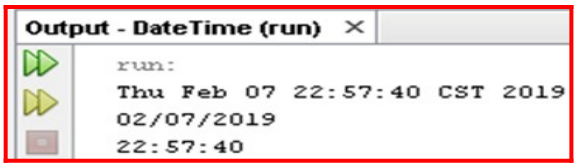
2) Save File as DateAndTime

Do Lab Exercise

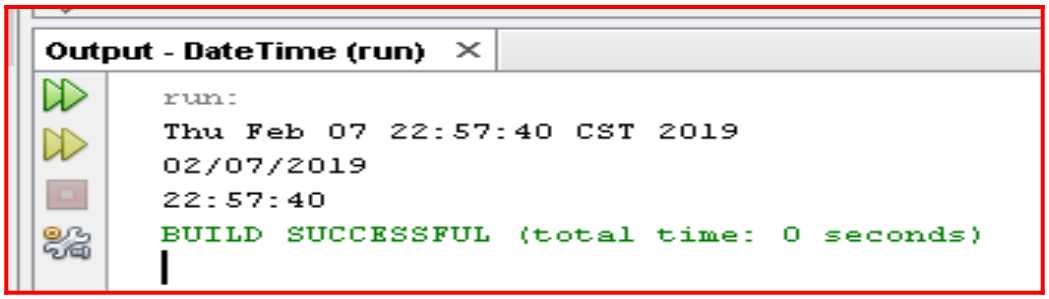
```
package dateandtime;
// 1) Use import statement to import the following libraries
import java.util.Date;
import java.text.SimpleDateFormat;
public class DateAndTime {
   public static void main(String[] args) {
     //1) Create an instance object called 'dateFormat' from class SimpleDateFormat
     SimpleDateFormat dateFormat = new SimpleDateFormat("MM/dd/yyyy");
     //2) Create an instance object called 'date' from the class Date
       Date date = new Date();

     // 3) Print out the date
     System.out.println(date);

     // 4) Print out the dateFormat
      System.out.println(dateFormat.format(date)); // prints in format 02/03/2019

     //5) Create an instance object called timeFormat from class SimpleDateFormat
      SimpleDateFormat timeFormat = new SimpleDateFormat("HH:mm:ss");
     // 4) Print out the timeFormat
      System.out.println(timeFormat.format(date));   // prints in format 22:57:40
```



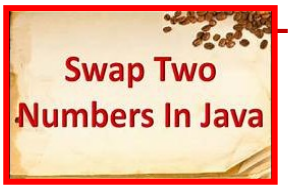
```
    }
}
```



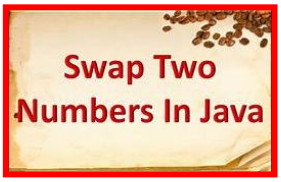
The Output of this Java project will look like the following:



How to Swap 2 Integer Numbers: By Using Another Temporary Variable (Do Lab Exercise 4) +++ 100 Points



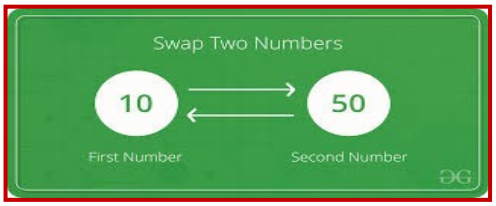
Do Lab Exercise 4



Lesson 82 Ex : How to Swap 2 Integer Numbers in Java language?

Problem or Project: Design and Code in Java Language the project to Swap 2 Integer Numbers Values.

The following Swap2Numbers Java Project will prompt the User to Enter 2 Numbers and then it will Swap the contents of First Number with the content of the Second Number. To do the Swapping you need to declare another variable called 'tempNumber' which is used as a Place Holder to hold the content value of First Number.



The Input/Output of the Java Swap project looks like the following:


```
Output - SwapStrings (run)
run:
Enter the First number: 22
Enter the Second number: 77

**** After Swapping the 2 Numbers ****

Value of the First number = 77
Value of Second number = 22
BUILD SUCCESSFUL (total time: 11 seconds)
```

1) Import the **Classes needed** in this project

import java.util.Scanner;

2. Declare the **Local integer Variables** used in this Java Swap project.

```
int tempNumber;
int firstNumber;
int secondNumber;
```

3) Create an **Object** called **'input'** from the **Class Scanner**

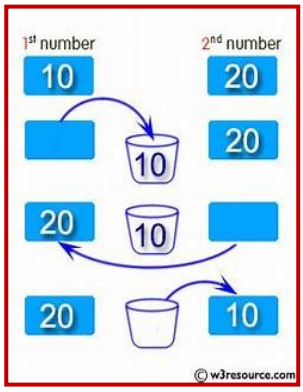
```
Scanner input = new Scanner(System.in);
```

4) Prompt the **user** to **Enter** the **First Number**

```
System.out.print("Enter the First Number: ");
```

5) Read or **Get** the **First Number** entered on keyboard and **Store** it in the **variable** called **firstNumber.**

```
firstNumber = input.nextInt();
```



6) Prompt the user to **Enter** the **Second Number**

```
System.out.print("Enter the Second Number: ");
```

7) Read or **Get** the **Second Number** entered on keyboard and **Store** it in the **variable** called **secondNumber**

```
secondNumber = input.nextInt();
```

8) Swap the **First Number** and **store** it in **tempNumber** variable

```
tempNumber = firstNumber;
```

9) Swap the **Second Number** and **store** it in **firstNumber** variable

```
firstNumber = secondNumber;
```

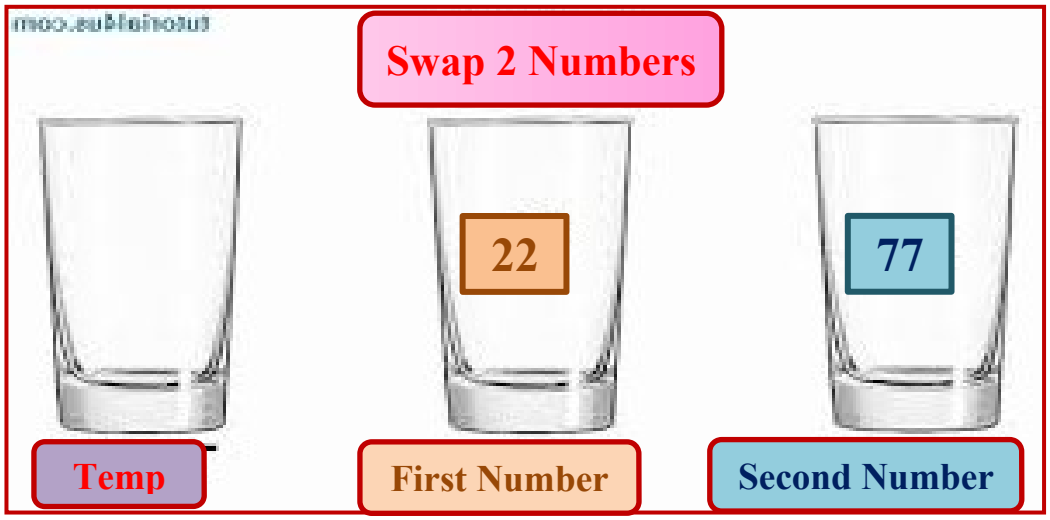
10) Swap Temp Number and **store** it i**n secondNumber** variable

```
secondNumber = tempNumber;
```

11) Print out the **title** and the **content** of the **Swapped Numbers**

```
System.out.println("\n** After Swapping the 2 Numbers **\n");
System.out.println("Value of the First number = " + firstNumber);
System.out.println("Value of Second number = " + secondNumber);
```

The following is a Graphic Description of Swapping 2 Integer variables:



	22	77
Temp Number	**First Number**	**Second Number**

// 8) **Swap** the **First Number** and **store** it in **tempNumber** variable

tempNumber = firstNumber; **22**

// 9) **Swap** the **Second Number** and **store** it in **firstNumber** variable

firstNumber = secondNumber; **77**

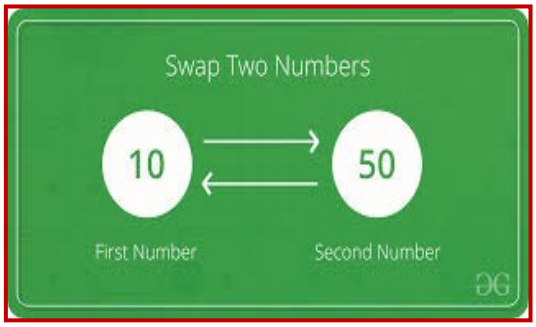
// 10) **Swap** the **Temp Number** and **store** it in **secondNumber** variable

secondNumber = tempNumber; **22**

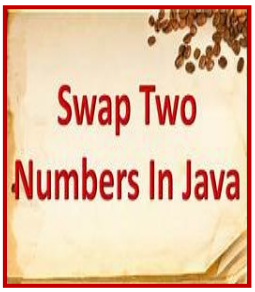
The Values stored in the 3 Variables are as follow:

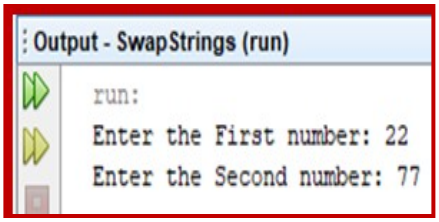
22	77	22
Temp Number	**First Number**	**Second Number**

1) Type the following project in NetBeans and
2) Save Java Project as **Swap2Numbers** **Do Lab Exercise**




```
package swap2numbers;

// 1) Import the Classes needed in this project
import java.util.Scanner;
/**
 * @author Ogar's Laptop
 */
public class Swap2Numbers {

   public static void main(String[] args) {

      // 2) Declare the Local integer variables used in this project
      int tempNumber;
      int firstNumber, secondNumber;

      // 3) Create an Object called 'input' from the Class Scanner

      Scanner input = new Scanner(System.in)

      // 4) Prompt the user to Enter the First Number
      System.out.print("Enter the First Number: ");

      // 5) Read or Get the First Number entered on keyboard and
      // Store it in the variable called firstNumber
      firstNumber = input.nextInt();

      // 6) Prompt the user to Enter the Second Number

      System.out.print("Enter the Second Number: ");

      // 7) Read or Get the Second Number entered on keyboard and
      // Store it in the variable called secondNumber
      secondNumber = input.nextInt();
```





```
        // 8) Swap the First Number and store it in tempNumber variable
        tempNumber = firstNumber;

        // 9) Swap the Second Number and store it in firstNumber variable
        firstNumber = secondNumber;

      // 10) Swap Temp Number tempNumber and store it in secondNumber variable
        secondNumber = tempNumber;

        // 11) Print out the title and the content of the Swapped Numbers
        System.out.println("\n**** After Swapping the 2 Numbers ****\n");
        System.out.println("Value of the First number = " + firstNumber);
        System.out.println("Value of Second number = " + secondNumber);

    }

}
```


```
**** After Swapping the 2 Numbers ****

Value of the First number = 77
Value of Second number = 22
BUILD SUCCESSFUL (total time: 11 seconds)
```

The Input/Output of the Java Swap project looks like the following:


```
Output - SwapStrings (run)
run:
Enter the First number: 22
Enter the Second number: 77

**** After Swapping the 2 Numbers ****

Value of the First number = 77
Value of Second number = 22
BUILD SUCCESSFUL (total time: 11 seconds)
```

Modify the Java Swap project to do the following:

1) **Modify the project to Swap 2 String Variables.**

2) **Prompt the user to Enter a Male Student First Name,**

Read the String Male First Name and Store it in a variable.

3) **Prompt the user to Enter a Female Student First Name, Read the String Female First Name and Store it in a variable.**

4) **Swap the Male First Name and the Female First Name and print them out as was done in the Swap 2 Numbers.**

5) **Add and Print a nice Header that consists of some lines.**

6) **Add and Print a nice Footer that consists of some lines and also print your Name as the programmer**.

Chapter 3 + Java Homework # 3 (Due Next Week) 200 Points

Name:_____ **CIS144 Java** Programming Language–Wright College

Answer the Following Chapter 03 Java Questions: *Instructor: Ogar Haji*

1) What is a **Program Language** and what is **Object-Oriented Language**?

2) Java is an **Object-Oriented Programming Language**. _____ (true/false)

3) **Java** is a **Case-Sensitive** and **Strongly-Typed language**. _____ (true/false)

4) What is an **Object**?

5) What is an **Object Oriented Programming (OOP), explain**?

6) What is a **Class**?

7) What is a **Encapsulation**?

8) What is **Polymorphism**?

9) What is **Inheritance**?

10) What is **Abstraction**?

11) **Java** is an **Object-Oriented Programming Language**. _____ (true/false)

12) **Objects** contain **properties** or **data** and **methods( )**. _____ (true/false)

13) What are **Properties**?

14) What are **Methods( )**?

15) **A Class** is the **Blueprint** or **Template** which you can use **to create** many **Objects**. _____ (true/false)

16) Why we make the **fields private** in a class?

17) Write the Java code to create a **Class** called "**House**" and **declare** some **fields** (**variables**) that a **house will** have.

18) Write the Java code to Define and code the **paint() method**.

19) What are the 4 **Access Modifiers** and what they do?

20) What does the **public** access modifier do?

21) What does the **private** access modifier do?

22) What does the **protected** access modifier do?

23) What does the **default** access modifier do?

24) What does **UML** stand for and what is **used for**?

25) Explain the following **UML Diagram**?

Top Compartment (Class Name)
Middle Compartment (Attributes or Fields)
Bottom Compartment (Methods or Actions)

26) Explain the following **UML Diagram**?

Applying **UML Diagram** for the **Class BankAccount**

BankAccount
- firstName : String (- means private) - lastName : String - balance : double

27) Explain the following **UML Diagram**?

+ setFirstName (firstName : String) + setLastName (lastName : String) + setBalance (balance : double)+ getFirstName : String (+ means public) + getLastName : String + getBalance : double

28) A **-** **sign** in front of **Attributes** or **Fields** means these **Fields** are _____.

29) A + sign in front of **Methods** means these **methods** are _____.

30) Write the Java code to **create** a **Class** called **BankAccount.**

31) Write the Java code to **declare** as **private** the **instance variable** or **field** called "**firstName**" of class **BankAccount.**

32) Write the Java code to Define and code **setFirstName()** method in the object

33) Write the Java code to Define and code **getFirstName()** method in the object

34) Write Java code to Create an object '**bankAccount1**' from **class BankAccount** in main().

35) Write the Java code to Print out the **initial value** of **first name** which is **'null'** by calling the **getFirstName()** method.

36) Keywords (Reserved Words) have **special meaning** to Java. You **Can Not use Keywords** as **variable Names**. Examples of **Keywords** are **int, private, public, if, double**. _____(true/false)

37) Code the following statements as **Comments** which are **ignored by Java Compiler**.

This project will Calculate the GrossPay of Employees.
The user will Enter the Hours Worked.
The User will also Enter the Hourly Rate.
Then User will click on Calculate Gross Pay Button to Calculate the GroosPay
Programmer: Instructor: Ogar Haji
Date-written: February 5, 2021

38) By Default, Java will **assign** a **Label** the **Text Property** of _____ and the **Name Property** of _____.

39) By Default, Java will **assign** a **Text Field** the **Text Property** of _____ and the **Name Property** of _____.

40) By Default, Java will assign a **Button** the **Text Property** of _____ and the **Name Property** of _____.

41) Indicate either **Valid** or **Invalid Control Names**. If **Invalid**, **what** is **wrong** with **Control's Name**.

txtHours

7txtDays

btnExit

void

strWhatis?#@YourName

lbl Gross Pay

42) Explain what are the **following Modes:**

a) **Design Time Mode**:

b) **Code Time Mode**:

c) **Run Time Mode**:

d) **Debug (Break) Mode**:

43) The **following 4 items should be identified** when defining what a Project is to do _____. (True/False).

1- **Purpose of the project:** what the program suppose to do
2- **Input:** what is the Input to the program?
3- **Processing & Calculations:** What Processing & Calculations the project will do on Input Data.
4- **Output:** What is the Desired Output and If it will be formatted.

44) What is a **RunTime Error**?

45) What is a **Logical Error**? What is **wrong** with **following statement** and **correct it.**

intAverage = intTest1 + intTest2 / 2;

.

Please, Read, Study and Practice the Lessons in the Java Handout

Chapter 3 + Java Lab Assignment #3A (Due Next Week) 100 Points

Name:_____ **CIS144 Java language + Wright College**

```
************ American Cars Rental *********
*****
******** Calculate Total Amount Due Program
Enter Customer Full Name: Mary Smith
Enter Number of Days Rented: 5
Enter Ending Miles: 800
Enter Beginning Miles: 700
*****************************************
```

Java Console Assignment #3A: American Cars Rental Project:

-+(Do Lab Assignment + Lab 3A)+-

Do Lab Assignment + Lab 3A

Problem or Project: Design and Code in Java Language the project to Calculate the Charges for American Cars Rental Company.

Do the 12 Must Steps to Design, Code & Solve a Project in Java Language

Do Steps 1 thru 7 in your Note Book or on Paper.

Note: When Modifying a Project, Do One Modification at a time.

1) Create a Console Application program first and Save it as AmericanCarsRentalConsoleLab3A.

2) Save Console project as AmericanCarsRentalConsoleLab3A

3) Note: Company Charges $40 Dollars per Day and 0.30 Cents per Mile.

4) The Calculations are as follows:

Charges By Days = Days Rented * 40

Charges By Miles = (Ending Miles – Beginning Miles) * 0.30

Total Charges = Charges By Days + Charges By Miles

Discount = Total Charges * 0.20

Amount Due = Total Charges – Discount

5) Run the project, and display Amount in Currency Formatting $.

6) Type the following Data: Your Name 5 800 700

1) **Write** a **Console Application** and **Save** it as **AmericanCars RentalConsoleLab3A**

The Input of the program will look like the following:

```
********** American Cars Rental ****************
*****                                      *******
******** Calculate Total Amount Due Program *********
Enter Customer Full Name: Mary Smith
Enter Number of Days Rented: 5
Enter Ending Miles: 800
Enter Beginning Miles: 700
*****************************************************
```

The calculated Output of the **project looks like the following:**

```
Charges By Days  = $200.00
Charges By Miles = $30.00
Total Charges    = $230.00
Discount (20%)   = $46.00
Total Amount Due = $184.00
*****************************************************
**********   End of Program    **********
****** Programmer: Instructor: Ogar Haji  *****
```

1) **Modify the Project**, **Add** a **Named Constant SALES_TAX** at **0.08** and **Calculate Sales Tax** and also **Total Amount Due**.
2) **Before** printing **Footers, Declare 2 integer variables** called **"number1"** and **"number2"** and **initialize number1** to **55** and **number2** to **33**.

3) Use Math.max() and Math.min() methods to find the Maximum number and the Minimum number of the 2 numbers and print them out.

Note: When Modifying a Project, Do One Modification at a time.

Chapter 3 + Java Lab Assignment #3B (Due Next Week) 100 Points

Name:_____ CIS144 Java language + Wright College

Chapter 3 + Lab 3B Java Project + NewHadra Pizza Palace Project (Due Next Week) 100 Points

Problem or Project: Design and Code in Java Language the project to Calculate and Manage Order Pizza in New Hadra Pizza Palace.

Do Steps 1 thru 7 in your Note Book or on Paper.

Do the 12 Must Steps to Design, Code and Solve a Problem in Java programming language.
To Design and Code a Console Java Project for "NewHadra Pizza Palace" project.
This is Chapter 3 Case 2 Study Java Project about a Pizza Store called "NewHadra Pizza Palace Store".

****

**** The NewHadra Pizza Palace ****
**** 6777 N. King Sargon Blvd., Chicago, IL 60645 ****
**** Phone: 1(773) 123-6777. ****
**** Store Hours: 10:00 AM till 2:00 AM ****

****

1) Go to the location where you saved the Java project NewHadraPizzaPalaceOH1 is located.
2) Coy the Folder by RC on Folder NewHadraPizzaPalaceOH1 and Click on Copy
3) RC on the Folder Project NewHadraPizzaPalaceOH1

4) Click on Paste to Paste and create a duplicate of the Java project.
5) Rename the Java project Folder to NewHadraPizzaPalaceOH2 (RC the Java project NewHadraPizzaPalaceOH1, C Rename, then type the new folder, just change 1 to 2 as: NewHadraPizzaPalaceOH2
6) We will Continue with the Previous Chapter 1 Case 1 project.
7) Get into NetBeans, Open Java Project NewHadraPizzaPalaceOH2

Modify the Java project "NewHadra Pizza Store" to do the following:

1) Comment out All the Print statement to Echo out the user inputs like, // System.out.println ("Your First Name is: " + firstName);
2) Also Comment out the prints out the letter of your First Name
3) Also Comment out the print out of Maximum and Minimum Values for Integer and Double.
3) Add a Main Menu after printing the Headings as follow:

```
**********************************************************
****** Please, Choose one of the following Options  *****
******                                              ******
***** 1) Enter 1 or 'S' for Small pizza             ******
***** 2) Enter 2 or 'M' for Medium pizza            ******
***** 3) Enter 3 or 'L' for Large pizza             ******
***** 4) Enter 4 or 'H' for Help and Redisplay Menu *****
**********************************************************
```

4) Declare **Constant Named variables** and use them to do the **calculations.** For example:
final double SMALL_PIZZA_PRICE = 10.00;
and also declare the following final double variables:
MEDIUM_PIZZA_PRICE = 12.99;
LARGE_PIZZA_PRICE = 15.99;

5) Prompt the **Customer** to **Enter his/her choice, read** the **String choice** and **save it** in a **String variable** called **"customerChoice".**
5) Use **if statement** to **Check Customer Choice: 1** is **Small Pizza** and **assign** the **correct constant Name** to it as shown below:

```
// Note: the || (pipe symbol) means 'or' and with 'or' only 1 condition
// may be true so the statement will be executed
```

```
if (customerChoice == "1" || customerChoice == "S" || customerChoice == "s") {
        pizzaPrice = SMALL_PIZZA_PRICE;
}
```

Repeat using **if** statement to **check for** the other **customer choices:**

6) Prompt the **Customer** to **Enter** his/her **Toppings: (**Choose from: **Extra Cheese (C), Onion (O)** or **Mushrooms (M), Read** the **customer Topping choice** and **save it** in a **String variable** called **"toppings".**

7) Declare **Constant Named variables** and **use** them to do the **calculations.** For example**:**

final double EXTRA_CHEESE_PRICE = 1.00;

And do the same for: ONIONS = 1.09, MUSHROOMS = 1.28.

8) Using **if statement, check** the **toppings** and **assign** the **values** as shown below**:**


```
if (toppings == "C" || toppings == "c") {
            toppingsPrice = EXTRA_CHEES_PRICE_;
}
```

Repeat using **if statement** to **check** for the other **toppings ("O"** for **Onions** and **"M"** for **Mushrooms:**

The Input/Output of the NewHadra Pizza Java project follows:

Output - NewHadraPizza (run)

```
run:
*****************************************************************
***** The New Hadra Pizza Palace ****************************
***** Address: 6777 N. King Sargon Street, Chicago, IL 60645 **
***** Phone is: 1(773) 123-6777.  ****************************
***** Store Hours are: 10:00 AM till 12:00AM    ***********
*****************************************************************
Enter your First Name: Ogar
Enter your Last Name: Haji
Enter your Cell Phone Number: 773-555-8888
*********************************************************
****** Please, Choose one of the following Options *****
******                                              ****
***** 1) Enter 1 or 'S' for Small pizza  ***********
***** 2) Enter 2 or 'M' for Medium pizza         ***********
***** 3) Enter 3 or 'L' for Large pizza  ***********
***** 4) Enter 4 or 'H' for Help and Redisplay Menu ****
*********************************************
```


```
Enter your Choice of Pizza(1 or 2 or 3: 1
Enter the Number of Pizza to Order: 2
Toppings available are:
'C' for Extra Cheese,'O' for Onion, 'M' for Mushrooms
Please Enter the Pizza Toppings: c
The Price of the Pizza is:$11.00
The Sub Total Pizza Amount is:$22.00
The Sales Tax Amount is:$2.20
The Total Amount Due is:$24.20
Enter the Customer Amount Tendered: 30
Your Amount Tendered is: $30.00
The Customer Amount Returned is:$5.80
Your Full Name is: Ogar Haji
************************************************************
******************** End of Job  **********************
** Programmer: Ogar Haji ++ Java Instructor (Your Name) ***
************************************************************
BUILD SUCCESSFUL (total time: 2 minutes 29 seconds)
```

Modify the NewHadra Pizza Project to do the following:

1) Add the following **before printing out** the **Footers**.

2) Declare an **integer variable** called **"intNumber"**.

3) Declare a **double variable** called **"doubleNumber"** and **assign** the **value "7.21"** to it.

4) Use **Assignment Operator** to **Assign** the **doubleNumber** to **intNumber** using **(int) Explicit Cast Operator**.

5) Print out the **value** of the **intNumber ("Value of intNumber = ")**.

6) Declare a **String variable** called **"lowerCaseFullName"** and **Convert** the **Value** of **fullName** to **lowercase** and **print** it out.

7) Declare a **String variable** called **"upperCaseFullName"** and **Convert** the **Value** of **fullName** to **uppercase** and **print** it out.

8) Swap the **two Strings** called **'lowerCaseFullName'** and **'upperCaseFullName** and **print** them out.

9) Declare a **String variable** called **"trimTheString"** and **assign** the **value** **"** **using trim to trim extra spaces** **"** from the **beginning** and **end** of the **String** and **print** it out**.**

10) Find out the **Length** of the **String "fullName"** and **print** it out**.**

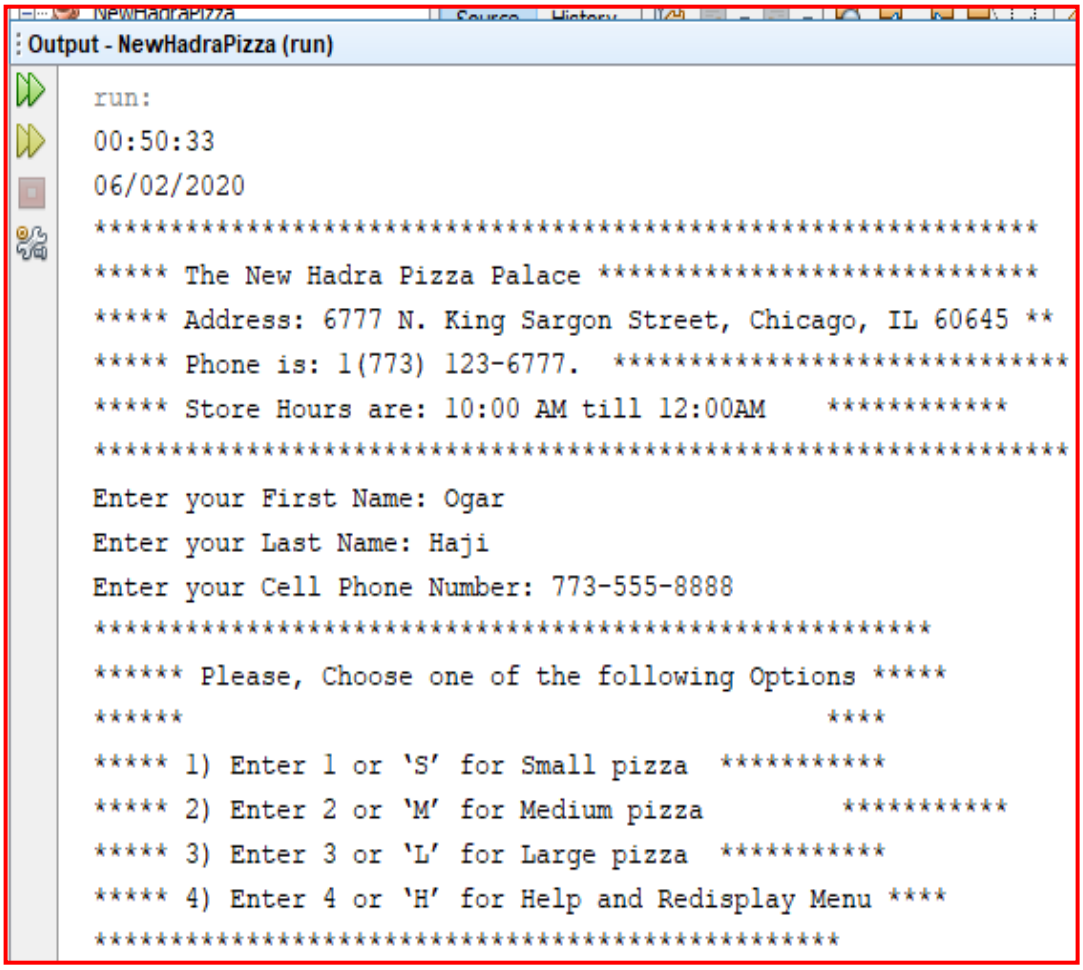
11) Find out **if** the **String "fullName" is Empty** and **print** it out.

12) Extract out the **SubString 'Your First Name'** from the **String "fullName"** and **print** it out.

13) Extract out the **SubString 'Your Last Name'** from the **String "fullName"** and **print** it out.

14) Insert Computer Date and **Time** at the **Top before printing** the **Headings**.

The Modified Input/Output of the NewHadra Pizza Java project is:




```
**************************************************
Enter your Choice of Pizza(1 or 2 or 3:) 1
Enter the Number of Pizza to Order: 2
Toppings available are:
'C' for Extra Cheese,'O' for Onion, 'M' for Mushrooms
Please Enter the Pizza Toppings: C
The Price of the Pizza is:$11.00
The Sub Total Pizza Amount is:$22.00
The Sales Tax Amount is:$2.20
The Total Amount Due is:$24.20
Enter the Customer Amount Tendered: 40
Your Amount Tendered is: $40.00
The Customer Amount Returned is:$15.80
Your Full Name is: Ogar Haji
The Value of intnumber = 7
The LowerCase Full Name is: ogar haji
The UpperCase Full Name is: OGAR HAJI
The trimmedString is: using trim to trim extra spaces
The Length of Full Name Ogar Haji is: 9
Is Full Name Empty Ogar Haji is: false
The substring First Name of Full Name Ogar Haji is: Ogar
The substring Last Name of Full Name Ogar Haji is: Haji
************************************************************
******************** End of Job  **********************
** Programmer: Ogar Haji ++ Java Instructor (Your Name) ***
************************************************************
BUILD SUCCESSFUL (total time: 56 seconds)
```

Java Instructor:
Ogar Haji

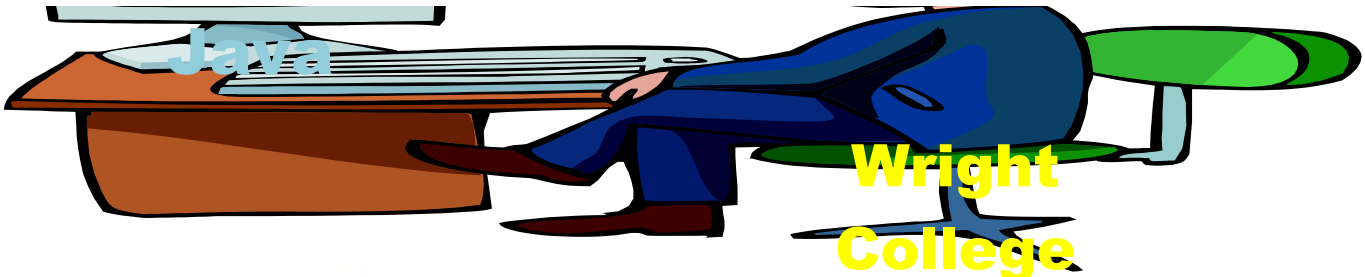
Note: Always **Upload** to **Brightspace 2 Files** of **same Java Project:**
1) The **Microsoft Word Document** of the **Java Project Code along with the Java Output Screen shots.**
2) The **Compressed** or **Zipped File** or **Folder** of **Java Project**.

Please, Read, Study and Practice the Lessons in the Java Handout

Wright College + Chapter 3

Object-Oriented Programming (OOP): Class, Object, Encapsulation, Polymorphism, Inheritance, and Abstraction

CIS 144 Java Programming Language– Introduction to Computer Programming



"Hands-On" Mastering Computer Logic, Design and Programming Using Java Language

Written By:

Ogar Haji

MS Computer Science

DePaul University ✚ Chicago, Illinois

Date Published: February 5, 2021