Introduction to Lab Manual

This lab manual aims at giving complete understanding of the core concepts of the topic and applying them in the exercises.

The lab manual consists of a set of lab exercises defined chapter wise. Each exercise has a definite objective defined. These objectives map with the terminal objectives defined at the beginning of each chapter. The problem statement is defined with clear instructions.

Some exercises have specific configuration or pre-condition mentioned. Advanced lab exercises are also given for some topics.

Appendix B given at the end of lab manual contains stepwise instructions to use eclipse IDE.

Configuration

jdk 1.6 should be installed. The lab exercises should be coded using Java language.

Specific configuration related to a particular exercise is mentioned. If it is not mentioned then the above configuration should be considered.

Structure of Lab Manual

Lab manual consists of different sections. The explanation of each structure is given below.

Objective

It states what you will achieve after completing a particular application. At the end of each lab exercise you should keep a track whether the objectives of that session are achieved.

Configuration

It is optional section and is present for specific lab exercises. If absent, then the configuration mentioned earlier should be considered.

Pre-condition

You should have understood the concepts explained in a particular chapter in the courseware thoroughly to solve the exercises mentioned.

But some lab exercises will have pre-condition explicitly mentioned.

Problem Statement

Problem statement for each lab exercise is given. It defines clear instructions to achieve the defined objective.

How to use the Lab Manual?

Whenever a programmer has to transform a problem statement into a program which a computer can execute, you should split the activity in the following manner:

* Read the problem statement carefully. Hint is given for some problem statements to help you to solve the problem.
* Preparation
* Decide the User interface (CUI or GUI) before hand.
* Write the algorithm or steps to be followed to solve the problem. You can also draw flowcharts if required.
* The program is made modular by writing functions. So pen down expected function prototypes and arguments on paper.
* Also decide how one module (function) would communicate with other module (function).
* Give a dry run to the algorithm written.
* Write the code.
* Execute the code.

**Coding practices**

The maintenance of code is easy if the code is written using coding practices. You should follow following coding practices while solving the lab exercises in this lab manual:

* Use meaningful names for variables, functions, file.
* Use uniform notation throughout your code.
* Code should be properly indented.
* Code should be commented. While documenting your code, write the purpose of your piece of code. What task is assigned to a method, what arguments are passed to it, what it returns should be clearly stated. Write a clear comment if you have added any statements for testing purpose.

Writing a right kind of well-documented software is an art along with your technical skills. Enough necessary documentation should be done. This makes the code easily maintainable.

|  |
| --- |
| Chapter 1 - Object-oriented Programming Concepts |

XYZ Organization has decided to go for computerization of its Payroll operations. First phase of automation will cover Human Resources, Accounts and Sales Departments. The structure of these departments is depicted below.

**Reporting Structure**

HR Executive🡪HR Manager🡪 Senior Manager

Salesman🡪Sales Manager🡪Chief Accountant

The Salary of employees consists of multiple heads:

|  |  |
| --- | --- |
| Income | Deductions |
| Basic Salary | Provident Fund |
| Dearness Allowance | Tax Deductible at Source |
| Housing Rent Allowance |  |
| Travel Allowance |  |

In addition to these, depending on the division of the employees, some more allowances are disbursed. Sales team gets commission, Accounts Team gets bonus and the HR team receives incentives. This extra disbursement is also dependent on their designation and position within the hierarchy. Additional gratuity amount is also paid to employees who have completed 5 years in the organization.

1. Identify the objects with their characteristics and behaviour to represent the problem statement.

2. Identify the object relationship representing the problem statement.

|  |
| --- |
| Chapter 2-Writing Java Classes |

Lab Exercise -1

Objective

* Construct a simple Java class: MyDate

Problem Statement

1. Construct a class Date which has attributes - day, month and year.Use initDate() method to print date.

Lab Exercise - 2

Objectives

* Create accessor and mutator methods.
* Create a constructor and overload it.

Pre-condition

* Date should be created.

Problem Statement 1

Create an object and initialize it using mutator methods and accesses it using accessor methods. Print the date.

Problem Statement 2

Create two objects and initialize them using no-argument and parameterized constructor respectively. Print the date.

Problem Statement 3

Consider that payroll software needs to be developed for computerization of operations of an ABC organization. The organization has employees.

* 1. Construct a class Employee with following members using private access specifies:
     1. Employee Id integer
     2. Employee Name string
     3. Basic Salary double
     4. HRA double
     5. Medical double
     6. PF double
     7. PT double
     8. Net Salary double
     9. Gross Salary double

Please use following expressions for calculations:

HRA = 50% of Basic Salary

PF = 12% of Basic Salary

PT = Rs. 200

* 1. Write no arguments constructor and parameterized constructor to initialize objects.
  2. Write properties for Employee Id, Employee Name, Basic Salary, Net Salary and Gross Salary.
  3. Write methods to display the details of an employee and calculate the gross and net salary.

Goss Salary = Basic Salary + HRA + Medical

Net Salary = Gross Salary – (PT + PF)

Lab Exercise - 3

Objective

* Overload methods

Problem Statement

Construct a class MathClass. Write overloaded method add() to add integers, float and string. [Overloaded add() for strings should implement concatenation].

Lab Exercise - 4

Objective

* Use of this keyword.

Pre-condition

* Date should be created.

Problem Statement

Write a method in the Date class that prints date using constructor chaining.

Lab Exercise - 5

Objective

* Use static variable and static method.

Pre-condition

* Employee class with attributes like employeeId, name, basic salary, etc. should be created.

Problem Statement

Consider the Employee class created in earlier lab session. Display the total number of employees using a static method, totalEmployees(). Modify the Employee class to implement auto-generation of employeeId.

Lab Exercise -6

Objective

* Implement toString() method.

Pre-condition

Employee class with attributes like employee id, name, basic salary, etc. should be created.

Problem Statement

Implement toString() method of Employee class to represent the string representation of employee object. The method should return employee id, employee name and gross salary as a string.

Lab Exercise - 7

Objectives

* Create user defined package and use it.
* Configure CLASSPATH variable.
* Use package access specifies.
* Identify the need of static import and use it.

Pre-condition

* Date should be created.

Problem Statement 1

Create 2 classes Student and Batch. Student class is in pack1 and Batch class is in pack2. Write a Test class to print Student and Batch information.

Problem Statement 2

Use the Student and Batch classes created earlier. It should contain public, protected, private and default attributes and using Batch class, check accessibility of these attributes.

|  |
| --- |
| Chapter 3 - Inheritance and Polymorphism |

Lab Exercise -8

Objective

* Construct "has-a" relationship between classes

Pre-condition

Employee and Date classes should be created.

Problem Statement

An employee has a date of joining. So define an object of Date class as a contained object in the Employee class. Make appropriate changes in the constructors and method that displays employee details.

Lab Exercise -9

Objectives

* Construct a hierarchy of objects in a given problem scenario.
* Override the methods of base class.

Pre-condition

Employee class with attributes like employee id, name, basic salary, etc. should be created.

Problem Statement

Construct a hierarchy of employees. Inherit class Manager and MarketingExecutive from base class Employee.

1. Manager class should have following members
2. Petrol Allowance : 8% of Basic Salary
3. Food Allowance : 13% of Basic Salary
4. Other Allowance : 3% of Basic Salary
5. MarketingExecutive class should have following members
6. Kilometers travelled
7. Tour Allowance: Rs 5/- per kilometer (Automatically generated).
8. Telephone Allowance: Rs.1500/-
9. Write constructors for the derived classes. [Hint: Use super keyword]
10. Override toString() method for the derived classes.
11. Override the methods for displaying details, calculating gross and net salary in Manager and MarketingExecutive class. Net Salary and PF calculation should not consider above allowances. Create objects of classes to invoke their respective methods.
12. Write a method called ShowDetails() in client code. This method should take a base class reference as a parameter. Invoke this method by passing objects of different classes as a parameter. The method should display the details of the employee and his gross and net salary using the base class reference. [Hint: Invoke the overridden methods using base class reference]

Lab Exercise -10

Objective

* Use methods from Object class: equals and hashCode.

Pre-condition

* Date class should be created.

Problem Statement

Create two instances of date and check whether these two dates are same or not, also check hashCode of those dates.

[Hint: use equals(Object obj) and hashCode() method.]

Lab Exercise -11

Objective

* Construct an abstract class and write abstract methods.

Problem Statement

Construct an abstract class Shape. Write an abstract method area() in the class Shape. Inherit the classes Circle and Rectangle from the class Shape. The method area() needs to be overridden by the derived classes.

1. Check if class Shape can be instantiated.
2. Check what happens if area() method is not overridden.
3. When area() method is overridden, create the objects of the derived classes and invoke the method.
4. Create a base class reference to invoke the method of respective class to which the reference is referring to.

Lab Exercise -12

Objective

* Define an interface and implement it in any class wherever it is required.

Pre-condition

* Employee, Date and Shape class should be created.

Problem Statement

Define an interface Printable with a method print(). Implement this interface in Employee, Shape and Date class.

Lab Exercise -13

Objective

* Create clone of the objects

Problem Statement

Clone the object of Date class by implementing Cloneable interface.

|  |
| --- |
| Chapter 4- Exception Handling |

Lab Exercise -14

Objectives

* To understand the concept of Exception and Handling techniques.
* To understand the concept of checked and unchecked exceptions.

Problem Statement 1

Write a class ‘FileDemo’ which accepts a file ‘file.txt’.If the file in not available then it will throw an exception ‘FileNotFoundException’ as well as close that file in a finally block.

Problem Statement 2

Create a class‘DivisionDemo’ .If user enters second number as a zero then throws an exception.

Hint: ArithmeticException

Note: Use throws

Lab Exercise - 15

Objective

* Use the concept of user defined exception.

Pre-condition

* Account class should be created.

Problem Statement

When person wants to withdraw money and if balance is not sufficient than throw an exception ‘LessBalanceException’.

Hint: minimum balance is 1000/-Rs.

|  |
| --- |
| Chapter 5- Collections |

Lab Exercise - 16

Objectives

* To understand the concept of Collection Interfaces.
* To understand the concept of NavigableMap and Set.

Pre-condition

* Employee class should be created.

Problem Statement

1. Add an attribute in Employee class.Attribute is: Set skillsets
2. Now write a class ‘UtilityList’ has a method printList().

This method collects all the employees and prints the entire list of employees.

1. Write a class ‘UtilityReport’ has a method showReport()
2. This method shows report like:

empName 🡪 salary (i.e. key -> value)

1. Write a class ‘SortUtil’that has a method sort().This method Sorts the records in ascending fashion. This class also has one more method sortBySalary() this method sorts records by salary.
2. Write a class ‘SearchUtil’ has a one method searchEmpInfo() which accepts which accepts the empNo as parameter and return employee information. If employee is not available than throw an exception ‘EmployeeRecordNotFoundException’.

Hint: UtilityList, UtilityReport, SortUtil, and SearchUtil all these four classes present inside a package: com.project.employees.util

Write generic definitions.

|  |
| --- |
| Chapter 6 - GUI with Java Swing Controls |

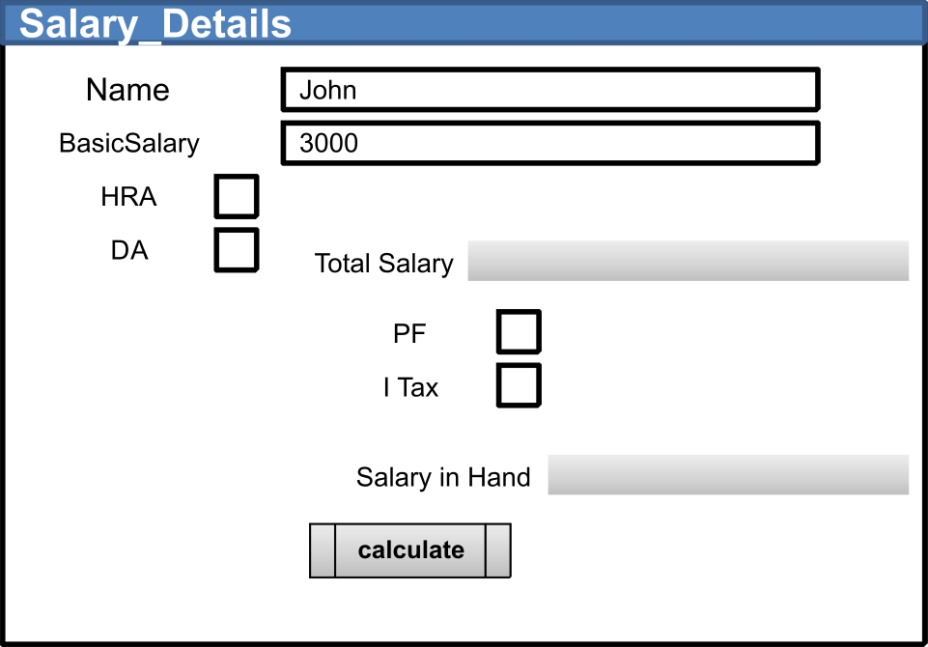
Lab Exercise - 17

Objective

* Develop an application to perform some operation on event triggered by a UI component.

Problem Statement 1

Create a GUI application to accept salary details from user like basic salary, HRA, DA, PF, I Tax to find gross salary and net salary.

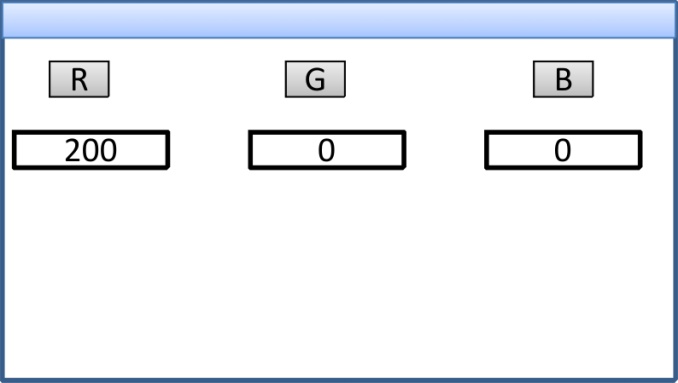


Salary calculation is based on following steps:

1. Enter basic salary of the employee.
2. Calculate HRA; HRA is 10% of the basic salary.
3. Calculate DA; DA is 15%of the basic salary.
4. Total Salary = BasicSalary + HRA + DA
5. Calculate PF;PF is 12% of Total Salary
6. Calculate I Tax; I Tax is 2% of Total Salary
7. When user of this system clicks on calculate button; net salary in Hand text field should show salary figure.
8. Salary in Hand=Total Salary-(PF + I Tax)
9. Net salary in hand text field is not editable.

Problem Statement 2

Create a GUI application which accepts three numbers from user and when user clicks on frame. It should show background color of JFrame.



1. Text field range is 0 to 255.Text field should not accept greater than 255 and less than 0.
2. Text field should not accept any character.

Hint: Use Mouse Event.

Expected Result:



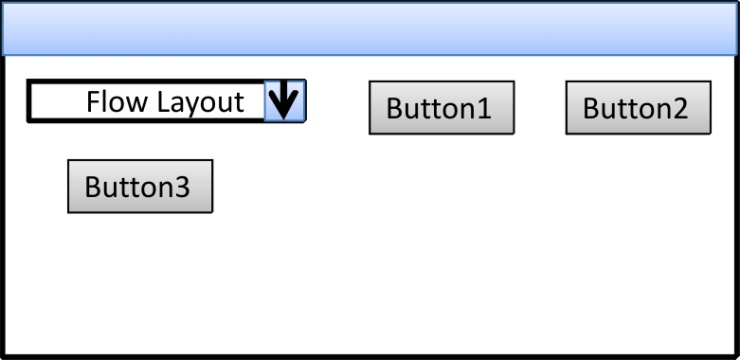
Lab Exercise - 18

Objective

* Arrange the components using simple layouts like Border, Grid and Flow layout.

Problem Statement

Create a GUI application which arranges same the components in Border Grid and Flow layout.



1. GUI frame should contain Combo box and 3 buttons.
2. When user selects ‘Flow Layout’ all the 4 components should be arranged in a FlowLayout.
3. When user selects ‘Grid Layout’ all the 4 components should be arranged in GridLayout.
4. When user selects ‘Border Layout’ all the 4 components should be arranged in BorderLayout.

Lab Exercise - 19

Objectives

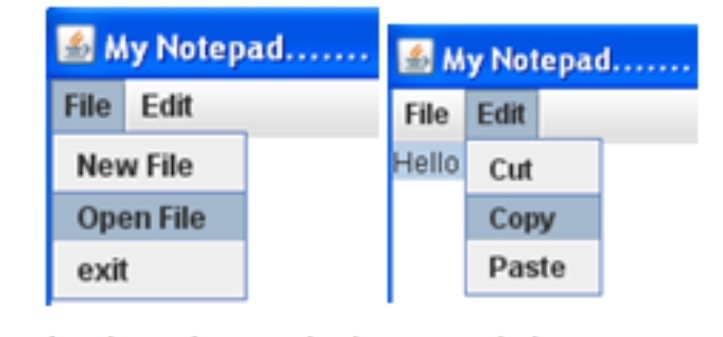
* Integrate built-in dialogs like File, Color Dialog etc. as part of the application (e.g. Text Editor).
* Use controls like Menu, Popup Menu and Dialog.

Problem Statement

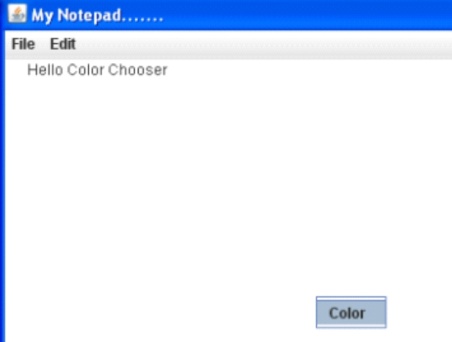
Design the Text Editor as shown below:



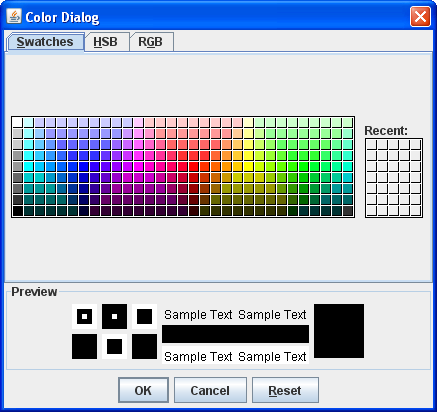
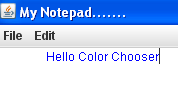
1. File, Edit menu should look like following:



1. The popup menu should be displayed on right button click. Popup menu displays option “Color”.



1. ColorChooser dialog should be displayed at the click of “Color” option.
2. After selecting the text from editor and colour from the dialog, text colour should be changed.

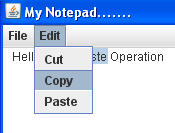
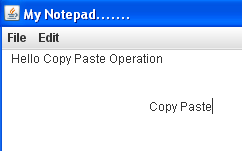
 

**Changed color**

1. Option ‘New’ should open a new client area.
2. Option ‘Open’ should display Open File Dialog box. The file should be opened in the client area.
3. Option ‘Exit’ should display dialog box. With ‘Yes, No, Cancel’ permission. If user selects ‘Yes’ then Notepad should be closed.



1. Option ‘Copy’ from Edit menu should enable copying of text into clipboard.
2. Option ‘Paste’ from Edit menu should enable copying text from clipboard.
3. Option ‘Cut’ from Edit menu should enable cutting text from the document and copying into clipboard.

After selecting paste option

|  |
| --- |
| Chapter 7 - Java Input-Output |

Lab Exercise - 20

Objective

* Perform file and directory operations.

Pre-condition:

* file1.txt and file2.txt should be present in a current directory with some pre-existing text data.

Problem Statement 1

Create an application to check availability of ‘file1.txt’.File information should also be retrieved.

Problem Statement 2

Read the contents from ‘file1.txt’ and overwrite the contents of ‘file2.txt’.

Problem Statement 3

Read and write file information using RandomAccessFile.

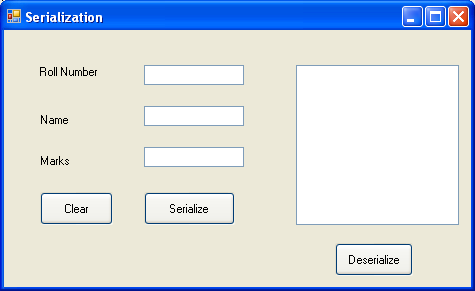
Lab Exercise - 21

Objective:

* Serialize the object using Object Serialization.
* **Pre-condition:**
* emp.txt should be present in a current directory.

Problem Statement

Design the user interface as shown below:



1. On clicking “Clear “button, the contents of all text boxes should be cleared and the focus should be set on the first text box.
2. On clicking “Serialize” button, the student object should be serialized to a binary file.
3. On clicking “Deserialize” button, the student object should be de-serialized and the details should be displayed in list box.

|  |
| --- |
| Chapter 8 - Java Sockets |

Lab Exercise - 22

Objective

* To create applications that will enable remote communication using Sockets.

Pre-condition:

* Client and Server application with UI creation is already given

Problem Statement

Create an application which accepts graphics like ovel, rectangle etc... When user draws such images on client window at a same time the same image should display on server screen.