**Problem Statement/ Case Study (If applicable)**

1. **Bank Account Management System:**
   * Funds Bank needs an application to feed new Account Holder information. AccountHolder will be a person. There are two types of accounts such as SavingsAccount, CurrentAccount.
2. **Employee Medical Insurance Scheme:**
   * By default, all employees in an organization will be assigned with a medical insurance scheme based on the salary range and designation of the employee. Refer the below given table to find the eligible insurance scheme specific to an employee.

|  |  |  |
| --- | --- | --- |
| **Salary** | **Designation** | **Insurance scheme** |
| >5000 and < 20000 | System Associate | Scheme C |
| >=20000 and <40000 | Programmer | Scheme B |
| >=40000 | Manager | Scheme A |
| <5000 | Clerk | No Scheme |

**Lab 2: Language Fundamentals, Classes and Objects**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  | At the end of this lab session, you will be able to: | |  |
|  | **Goals** |  |  Write a Java program that displays person details | |  |
|  |  |  | Working with Conditional Statements |  |
|  |  |  |  |  |
|  |  |  |  Create Classes and Objects | |  |
|  | **Time** |  | 120 minutes | |  |
|  |  |  |  |  |  |

2.1 Write a java program to print person details in the format as shown below:

Person Details:

\_\_\_\_\_\_\_\_\_\_\_\_

First Name: Divya

Last Name: Bharathi

Gender: F

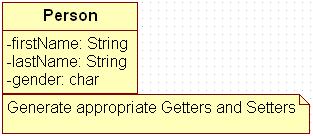
Age: 20

Weight: 85.55

**Figure 12: Sample output of Person details**

2.2: Write a program to accept a number from user as a command line argument and check whether the given number is positive or negative number.

2.3: Refer the class diagram given below and createa personclass.



**Figure 13: Class Diagram of Person**

Create default and parameterized constructor for Person class.

Also Create “PersonMain.java” program and write code for following operations:

1. Create an object of Person class and specify person details through constructor.
2. Display the details in the format given in Lab assignment 2.1

2.4: Modify Lab assignment 2.3 to accept phone number of a person. Create a newmethod to implement the same and also define method for displaying persondetails.

2.5: Modify the above program, to accept only ‘M’ or ‘F’ as gender field values. Use Enumeration for implementing the same.

**Lab 3: Exploring Basic Java Class Libraries**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | At the end of this lab session, you will be able to: |  |
|  | **Goals** |  |  Working with Basic Java Class Libraries |  |
|  |  |  Working with Strings and Date and Time API |  |
|  |  |  |  |
|  | **Time** |  | 100 minutes |  |
|  |  |  |  |  |

3.1: Create a method which can perform a particular String operation based on the user’s choice.

The method should accept the String object and the user’s choice and return the output of the

operation.

Options are

* Add the String to itself
* Replace odd positions with #
* Remove duplicate characters in the String
* Change odd characters to upper case

3.2: Create a method that accepts a String and checks if it is a positive string. A string is considered a positive string, if on moving from left to right each character in the String comes after the previous characters in the Alphabetical order.For Example: ANT is a positive String (Since T comes after N and N comes after A). The method should return true if the entered string is positive.

3.3: Create a method to accept date and print the duration in days, months and years with regards to current system date.

3.4: Revise exercise 3.3 to accept two LocalDates and print the duration between dates in days, months and years.

3.5: Create a method to accept product purchase date and warrantee period (in terms of months and years). Print the date on which warrantee of product expires.

3.6: Create a method which accept zone id and print the current date and time with respect to given zone. (Hint: Few zones to test your code. America/New\_York, Europe/London, Asia/Tokyo, US/Pacific, Africa/Cairo, Australia/Sydney etc.)

3.7: Modify Lab assignment 2.3 to perform following functionalities:

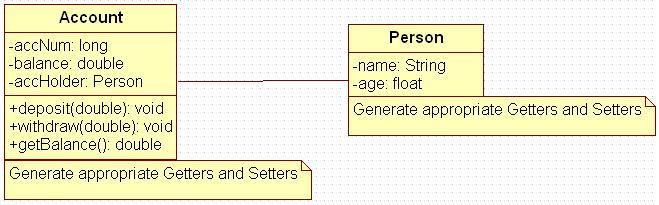
1. Add a method called calculateAge which should accept person’s date of birth and calculate age of a person.
2. Add a method called getFullName(String firstName, String lastName) which should return full name of a person

Display person details with age and fullname.

**Lab 4: Inheritance and Polymorphism**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | At the end of this lab session, you will be able to: |  |
|  | **Goals** |  |  Write a Java program that manipulates person details |  |
|  |  |  Working with Inheritance, Polymorphism |  |
|  |  |  |  |
|  | **Time** |  | 120 minutes |  |
|  |  |  |  |  |

4.1: Refer the case study 1in Page No: 5 and create Account Class as shown below in class diagram. Ensure minimum balance of INR 500 in a bank account is available.





1. Create Account for smith with initial balance as INR 2000 and for Kathy with initial balance as 3000.(accNum should be auto generated).
2. Deposit 2000 INR to smith account.
3. Withdraw 2000 INR from Kathy account.
4. Display updated balances in both the account.
5. Generate toString() method.

4.2: Extend the functionality through Inheritanceand polymorphism (Maintenance)

Inherit two classes Savings Account and Current Account from account class. Implement the following in the respective classes.

1. Savings Account
   1. Add a variable called minimum Balance and assign final modifier.
   2. Override method called withdraw (This method should check for minimum balance and allow withdraw to happen)
2. Current Account
   1. Add a variable called overdraft Limit
   2. Overridemethod called withdraw (checks whether overdraft limit is reached and returns a boolean value accordingly)

**Lab 5: Abstract classes and Interfaces**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Goals** |  | At the end of this lab session, you will be able to: |  |
|  |  |  Use of abstract classes and interfaces |  |
|  |  |  |  |
|  | **Time** |  | 90 minutes |  |
|  |  |  |  |  |

5.1: Refer the case study 2 in page no: 5 and create an application for that requirement by creating packages and classes as given below:

**a) com.mp.eis.bean**

In this package, create “Employee” class with different attributes such as id, name, salary, designation, insuranceScheme.

**b) com.mp.eis.service**

This package will contain code for services offered in Employee Insurance System. The service class will have one EmployeeService Interface and its corresponding implementation class.

**c)** **com.mp.eis.pl**

This package will contain code for getting input from user, produce expected output to the user and invoke services offered by the system.

The services offered by this application currently are:

1. Get employee details from user.
2. Find the insurance scheme for an employee based on salary and

designation.

1. Display all the details of an employee.

5.2: Use overrides annotation for the overridden methods available in a derived class of an interface of all the assignments.

5.3: Refer the problem statement 4.1. Modify account class as abstract class and declare withdraw method.