GEDİZUNIVERSITY

Faculty of Engineering and Architecture

Computer Engineering Department

COM 102 – OBJECT ORIENTED PROGRAMMING
DURING LAB Assignment#2

Academic Year: Spring 2016
Date : March 15,2016
Course Instructor: Assoc. Prof. Dr. I.Furkan INCE
Course Assistant: Res.Asst. Arzum KARATAŞ & Res.Asst. Feyza GALİP

1- Create a class named “CheckingBankAccount” that includes three instance variables – an account number (type String), a name of account holder (type String) and a current balance (type double). Then do the following tasks.

- Draw a UML class diagram for the classes.
- Write a constructor method that initializes the three instance variables. (Do not assume that the argument values are always correct and check the input validity.)
- Write set and get methods for each instance variable.
- Write a method that accepts deposit (e.g., it adds an amount (in Turkish Liras, TL) to the current balance.)
- Write a method that process a check. (e.g., it subtracts the amount of the check from the current balance. Overdrafts are allowed, so the balance can become negative. However, if the balance is less than 1000.00TL before the check is processed, 5.5 TL is charged for each check.)
- Write a method that shows account number, account holder and current balance.
- Write a test application named CheckingBankAccountTest that demonstrates class CheckingBankAccount’s capabilities (e.g. constructor, set, get and the other methods)

2- Create a class called “Employee” that includes six instance variables – a first name (type String), and a last name (type String), an age (type int), an employee position (type String) and work hours for each week (type int) and a payment for each work hour(double). Then do the following tasks.

- Draw a UML class diagram for the class and its Test class.
- Write a constructor that initializes the six instance variables. Be careful when you initializing these variables. That is, do a validation for inputs (e.g. first name, last name, employee position cannot be null, age should be greater than 16 and less

than 70, work hour for each week should be positive and price per hour should be non-negative.)

- Write set and get methods for each instance variable. Do the validity check, also.
- Write a calculateWeeklySalary method that calculates salary of employee (salary = work hours* price for an hour).
- Write a displayEmployeeInformation method that displays first name, last name, age, employee position and salary of employee.
- Write a test application named EmployeeTest that demonstrates class Employee capabilities. Create three Employee objects and display each employee information. Then, calculate average salary and inform the user.