Arab Academy of Science and Technology and Maritime Transport

COLLEGE OF ENGINEERING & TECHNOLOGY

Computer Engineering Department

Course: **Object Oriented Programming**

Course Code: **CC316**

Sheet No.: **1**

1. What are: syntax errors (compile errors), runtime errors, and logic errors?
2. Identify and fix the errors in the following code:

```
public class Welcome {
        public void Main(String[] args) {
                System.out.println('Welcome to Java!);
        }
}
```

3. What are the differences between constructors and methods?
4. What is the output of the following code?

```
public class A {
boolean x;
public static void main(String[] args) {
        A a = new A();
        System.out.println(a.x);
        }
}
```

5. Suppose that the class F is defined in (a). Let f be an instance of F. Which of the statements in (b) are correct?

```
a)
public class F {
 int i;
 static String s;
 void imethod() {
 }
 static void smethod() {
 }

}
```

```
b)
System.out.println(f.i);
System.out.println(f.s);
f.imethod();
f.smethod();
System.out.println(F.i);
System.out.println(F.s);
F.imethod();
F.smethod();
```

6. Can you invoke an instance method or reference an instance variable from a static method? Can you invoke a static method or reference a static variable from an instance method? What is wrong in the following code?

```
public class C {
public static void main(String[] args) {
method1();
}
public static void method1() {
        method2();
}
public static void method2() {
        System.out.println("What is radius " + c.getRadius());
}
Circle c = new Circle();
}
```

7. What is an accessor method? What is a mutator method? What are the naming conventions for accessor methods and mutator methods?
8. What are the benefits of data field encapsulation?
9. What is wrong in the following code?

```
public class Test {
        private int id;
        public void m1() {
                this.id = 45;
        }
        public void m2() {
                Test.id = 45;
        }
}
```

10. (*The Account class*) Design a class named **Account** that contains:

- A private **int** data field named **id** for the account (default **0**).
- A private **double** data field named **balance** for the account (default **0**).
- A private **double** data field named **annualInterestRate** that stores the current interest rate (default **0**). Assume all accounts have the same interest rate.
- A private **Date** data field named **dateCreated** that stores the date when the account was created.
- A no-arg constructor that creates a default account.
- A constructor that creates an account with the specified id and initial balance.
- The accessor and mutator methods for **id**, **balance**, and **annualInterestRate**.
- The accessor method for **dateCreated**.
- A method named **getMonthlyInterestRate()** that returns the monthly interest rate.
- A method named **getMonthlyInterest()** that returns the monthly interest.
- A method named **withdraw** that withdraws a specified amount from the account.
- A method named **deposit** that deposits a specified amount to the account. Draw the UML diagram for the class and then implement the class. (*Hint*: The method **getMonthlyInterest()** is to return monthly interest, not the interest rate. Monthly interest is **balance * monthlyInterestRate**. **monthlyInterestRate** is **annualInterestRate / 12**. Note that **annualInterestRate** is a percentage, e.g., like 4.5%. You need to divide it by 100.) Write a test program that creates an **Account** object with an account ID of 1122, a balance of $20,000, and an annual interest rate of 4.5%. Use the **withdraw** method to withdraw $2,500, use the **deposit** method to deposit $3,000, and print the balance, the monthly interest, and the date when this account was created.