The University Of Jordan
Faculty of Engineering and Technology
Computer Engineering Department

Object-oriented Engineering Problem Solving Laboratory

Lab 1: Java Language Essentials I
Instructor: Dr. Khalid A. Darabkh

Name: ……, **ID:** ……, **Section Number:** ……

In the Desktop of assigned PC, create a folder that carries your name in which a subfolder that represents in-lab work has to be created in every lab. However, this lab contains four programming tasks. You need to handle them properly.

Experiment 1: Write the following code and save it in a file called *Experiment1.java*. However, there is a lot of compilation errors found out there. You are required to fix them and get the program run successfully.

```
class Experiment1
{
public static void main (String []arg)
   {
       int i=5;
       Boolean b_$5=6;
       int byte=5;
       Short $4=12;
       double _3%=33.33;
       float f=3.5;
       long L66=123456789;
       system.out.println(i);
   }
}
public class Test1
{
public static void main (String []arg)
   {
       system.out.println("Welcome to Test1");
   }
}
```

Experiment 2: You are required to write the following code and save it in a file called ***Experiment2.java***. Furthermore, in the space provided, you need to insert only one statement to call the executable function of class ***Test2***.

```
class Experiment2
{
  public static void main (String [] arg)
   {
   // A missing statement
   }
}
class Test2
  {
    public static void main (String []arg)
   {
    System.out.println("Welcome to Test2");
   }
}
```

Experiment 3: You need to create a Java file, named as ***Experiment3.java***, along with appropriate code for the purpose of computing the following equation:

$$F = ((4 * 5 - 2) + 33 - 7 * 4 - (11\%3))/ 2$$

Experiment 4: You should create a Java program that consists of three classes, named as ***Experiment4***, ***Test3***, and ***Test4***. The executable class is ***Experiment4***. Inside class ***Test3***, declare and initialize three variables ***i,j,*** and ***k*** of type ***byte***, ***short***, and ***int*** and have the values of 12, 222, and 1240, respectively. Moreover, define a method called ***Method1*** that accepts two short variables (s and m) and returns a byte according to the following expression:

$$i=i*m+s+j-k$$

In class ***Test4***, just create an object of type ***Test3***. In the executable function, display the following in sequence through only class ***Test4***:

1. The sum of i, j, and k
2. The return value after calling ***Method1 and*** passing to it the values of 33 and 44.

The End: Good Luck!

The University Of Jordan
Faculty of Engineering and Technology
Computer Engineering Department

Object-oriented Engineering Problem Solving Laboratory

Lab 2: Java Language Essentials II
Instructor: Dr. Khalid A. Darabkh

Name: ……, **ID:** ……, **Section Number:** ……

This lab contains four programming tasks. You need to handle them properly.

Experiment 1:

1. Create a Java file and save it as "Experiment1.java". In this file, declare a class called "Conversion" in which two methods are to be defined as follows:

    - "Celsius" which accepts a double number, identified by *F* to represent a Fahrenheit temperature, and returns the corresponding Celsius value based on the following formula:

    $$C = 5.0 / 9.0 * (F - 32)$$

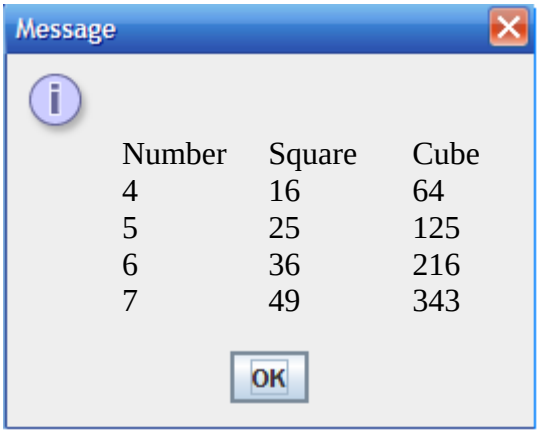
    - "Fahrenheit" which accepts a double number, identified by *C*; referring to Celsius temperature, and returns the corresponding Fahrenheit value based on the following formula:

    $$F = 9.0 / 5.0 * C + 32$$

2. Declare another class called "Experiment1.java". In this class, enable the user to enter either a Fahrenheit or Celsius value and accordingly call the appropriate function, of the above, to display the conversion utilizing "JOptionPane" class.

Experiment 2: Create a Java file and name it "Experiment2.java". In this file, the user should insert any month of a year in integer format to be accordingly displayed in *String* format. For example, if the user inserts "9", then the string "September" has to be displayed.

Experiment 3: Create a Java file and name it "Experiment3.java". In this file, two integer numbers, which represent an integer interval, have to be read using "Scanner" class. For each integer number, falls in this interval, calculate its square and cube values and accordingly display the results on a message dialog as shown below, assuming the insertion was the pair of (4,7).



Experiment 4: Create a Java file and name it "Experiment4.java". In this file, you need to write the appropriate code to find the largest number among three inserted ones. The main task, which refers to defining just one method that accepts only two numbers, should be done in a class called *Test1*.

The End: Good Luck!

The University Of Jordan
Faculty of Engineering and Technology
Computer Engineering Department

Object-oriented Engineering Problem Solving Laboratory

Lab 3: Operators, Arithmetic Promotion, Method Calling, and Shallow Polymorphism
Instructor: Dr. Khalid A. Darabkh

Name: ……, **ID:** ……, **Section Number:** ……

This lab contains five programming tasks. You need to handle them all properly.

Experiment 1: In this experiment, you will practice the use of automatic and member variables. Therefore, in the following code, correct wherever required to get it successfully complied.

```
public class Experiment1
{
   public static void main(String []a)
    {
     System.out.println("Experiment#1");
    }
}
class Test1{
   short i;
   int method1(byte b)
     {
      byte c;
      return b*i+c;
     }
}
```

Experiment 2: This experiment concerns about practicing call by value and reference approach. Hence, you should code the following in sequence:

1. **Create an executable class under the name of "Experiment2A.java". In this class, define the main method along with a local integer variable**

which is identified by "number" and initialized by 5.

2. **Create another class and name it "Experiment2B.java". In this class, write a method called ("method1") which accepts an integer number and returns that number after adding 2 to its value.**
3. **In the executable function, create an instance of class "Experiment2B". Thereafter, call "method1" and pass to it the variable ("number") as an argument.**
4. **Display the value returned from the method and value of variable ("number").**
5. **Write down your observations:**

   ----

6. **To this extent, define a new class called "Experiment2C.java" that has an integer member identified by "myNumber" and initialized by 10.**
7. **Back to class "Experiment2B", define a method called "method2" which basically accepts an argument of type "Experiment2C" and changes the value of "myNumber" to 100.**
8. **In the executable function, create an object of type "Experiment2C" for the reason of calling "method2", found in class "Experiment2B".**
9. **Display the value of "myNumber", found in class "Experiment2C".**

10. **Write down your conclusions:**

Experiment 3: In this experiment, you need to substitute the code implemented below with a shorter one that uses a ternary operator.

```
public class Experiment3
{
public static void main(String []a)
     {
        long k,x=56012;
        long y=46012;
        if (x==y)  k=x;
        else k=x-y;
        System.out.println(k);
     }
}
```

Experiment 4: This experiment concerns about *primitive conversion: arithmetic promotion.* Therefore, in the following code, correct wherever required to get it successfully complied.

```
class Experiment4
{
public static void main(String []a)
   {
     byte b=12;
     short s=13*b;
     float f =44.5F;
     long l=f*s;
     b+=6;
     s=s+5;
   }
}
```

Experiment 5: In this experiment, you will practice the use of method overloading. Hence, in the following code, correct wherever required to get it successfully complied

```
class Experiment5
{
public static void main(String []a)
{
  Test ref=new Test();
  System.out.println(ref.min(5,4));
  System.out.println(ref.min(5F,4F));
  System.out.println(ref.min(5L,4L));
  System.out.println(ref.min(5,4));
}
}
class Test
{
 int min (int i, int j)
 {
        if (i>j) return j;
        else return i;
 }
 float min (float i, float j)
  {
        if (i>j) return j;
        else return i;
  }
 long min (long i, long j)
  {
        if (i>j) return j;
        else return i;
  }
 int min (int i, int j)
  {
        if (i>j) return j;
        else return i;
  }

}
```

The End: Good Luck!

The University Of Jordan
Faculty of Engineering and Technology
Computer Engineering Department

Object-oriented Engineering Problem Solving Laboratory
Instructor: Dr. Khalid A. Darabkh

Lab 4: Dealing with Arrays

Name: ……, **ID:** ……, **Section Number:** ……

Consider the following to be defined in the executable function of a class called *ArraysDemo*:

A, B, and k

where ***A*** and ***B*** are vectors of the same size while ***k*** is a certain constant.

D

where ***D*** is a two-dimensional array.

1. **In the executable function, you need to code the following:**

   A. The size of arrays (***A***,***B***) and constant ***k*** should be given interactively through the keyboard. Moreover, arrays' elements should be initialized randomly in the range of [10-109].

   B. The dimensions of matrix ***D*** should be equal and given interactively through the keyboard while its elements should be initialized according to the row-index to the power of column-index formula for each element.

2. **You are required to code the following in a class called *ArraysProc* (Not executable class):**

   A. In a method called ***Find***, the expression (***A.B+k***) must be evaluated. In the executable function, the resulting array must be named as ***C*** and accordingly displayed.

   B. In a method called ***Max***, the largest element of array ***C*** must be found and interactively displayed in the executable function.

   C. In a method called ***Sort***, the array ***C*** must be sorted into ascending numerical order. Additionally, the sorted ***C*** array must be displayed in the executable function.

   D. In a method called ***Reverse***, all elements of array ***C*** must be totally reversed and interactively displayed in the executable function.

   E. In a method called ***Copy***, the last three elements of array ***C*** have to be copied to the first three elements of array ***B*** without using loops. Array ***B*** should be interactively displayed in the executable function.

   F. In a method called ***maxDiagonal,*** the largest element found in both diagonals must be found and interactively displayed in the executable function.

The End: Good Luck!

The University Of Jordan
Faculty of Engineering and Technology
Computer Engineering Department

Object-oriented Engineering Problem Solving Laboratory
Instructor: Dr. Khalid A. Darabkh

Lab 5: Classes: String and Math

Name: ……, **ID:** ……, **Section Number:** ……

Consider the following to be defined in the executable function of a class called *StringMathDemo*:

A,

where ***A*** is a reference to a one-dimensional array of type *String*.

1. **In the executable function, you need to code the following:**
   A. The size and elements of array ***A*** should be given interactively through the keyboard.
   B. The following expression should be evaluated: $\tan(\pi/3)+\sqrt[4]{|(x+y)^3|}$, where x and y are integer numbers that should be inserted through the keyboard.
2. **You are required to code the following in a class called *StringMathProc* (Not executable class):**
   A. In a method called ***getSubString***, a substring should be returned and accordingly displayed in the executable function. This function accepts a string from array ***A*** along with two integer numbers that represent begin and end indices.

B. In a method called ***DuplicPercentage***, the percentage of only duplicate characters, found in a certain input string from array ***A***, should be evaluated and accordingly displayed along with these duplicate characters. This method is expected to return nothing.

C. In a method called ***charCount***, the number of only characters (excluding digits), found in a certain input string from array ***A***, is expected to be evaluated and returned to be displayed in the executable function.

D. In a method called ***Sort***, the array ***A*** must be sorted into alphabetical order. Additionally, the sorted ***A*** array must be displayed in the executable function.

The End: Good Luck!

The University Of Jordan
Faculty of Engineering and Technology
Computer Engineering Department

Object-oriented Engineering Problem Solving Laboratory
Instructor: Dr. Khalid A. Darabkh

Lab 6: Inheritance and Polymorphism

Name: ……, **ID:** ……, **Section Number:** ……

In a certain bank, there are a lot of accounts that can be created for valuable customers. However, assume the following accounts are available: Checking, Saving, Interest Bearing Account, as well as Brokerage. The important entries (either attributes or operations) of each account, mentioned above, are described as below:

Checking Account	Saving Account	Interest Bearing Account:	Brokerage Account:
• Name • Account Number • Set up Date • Status :OPEN/CLOSE • Balance • Description • Bank Name • Fee • Deposit • Withdraw	• Name • Account Number • Set up Date • Status • Balance • Description • Bank Name • Savings Rate • Deposit • Withdraw	• Name • Account Number • Set up Date • Status • Balance • Description • Bank Name • Annual Interest Rate • Add Monthly Interest	• Name • Account Number • Set up Date • Status • Balance • Description • Bank Name • Brokerage Fees per Trade • Trade Amount • Trade Profit • Do Trade Transaction

What are the steps required?

You are required to build up an appropriate design to organize these financial accounts using the concept of *Object-oriented Java Programming*. In your demonstration, you need to create at least an account for each type and accordingly pass proper information. After that, you need to request different relevant services (such as *deposit*, *withdraw*, *add monthly interest*, and *do trade transaction*) to the accounts' objects created previously. Lastly, you need to show the end balance, after performing the transactions, along with some suitable customer information.

Hints:

1. The parameters of *Do Trade Transaction* are the *trade amount*, *brokerage fees*, as well as *trade profit*.
2. The parameter of *Add monthly Interest* is a certain annual interest rate
3. The return value of all requested services is always the balance.
4. Before requesting any service, make sure the account is still valid (i.e., still open).
5. The *description* attribute of all aforementioned accounts belongs to detailed customer information.

Good Luck!

The University Of Jordan
Faculty of Engineering and Technology
Computer Engineering Department

Object-oriented Engineering Problem Solving Laboratory
Instructor: Dr. Khalid A. Darabkh

Lab 7: Garbage collection, packaging, access Modifiers, as well as static and abstract modifiers

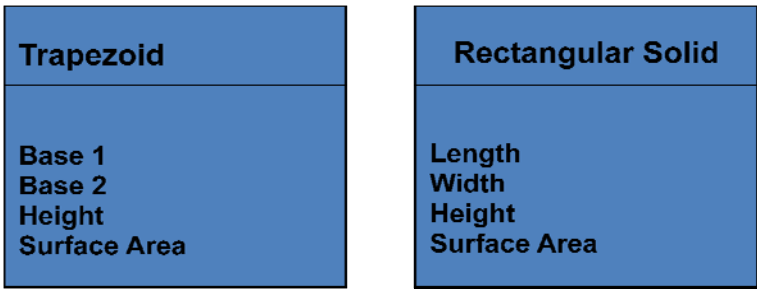
Name: ……, **ID:** ……, **Section Number:** ……

Experiment #1:

Write a java code to satisfy the following conditions:

1. Create three objects of type *GarbageDemo* where each mainly contains:
   - Two integer states in which one of them is static whereas both of them should be dynamically initialized.
   - A behavior called *"processing"* that mainly has no signature and modifies both states by adding the non-static state to itself and multiplying the static state by itself.
   - A behavior called *"finalize"* which is inherited from Object class and implemented through providing printing statement(s) indicating its main task.
2. Make the reference of the first object be equal to the one pointing to the third object.
3. Make the reference of the last object point to nothing
4. Create an object of type *GarbageTest,* that has no states, and invoke its behavior called "check". As a signature, the behavior accepts a reference to an object of type *GarbageDemo*. As implementation, this behavior creates a new object of type *GarbageDemo*.
5. What is output after considering the previous steps? What is your observation about the static state?
6. Can we show the contents of the *finalize* method without explicitly calling it? How?

Experiment #2:



You are required in this experiment to construct an appropriate design to the above entities to implement the abstract modifier and the concept of deep polymorphism. After that, create two objects of type **"Trapezoid"** and two more objects of type **"Rectangular Solid"**. The target of instantiating these objects is to find the total surface area.

Notes:

1. Surface area of **Trapezoid is** $\frac{Base1 + Base2}{2} Height$
2. Surface area of **Rectangular Solid is** $2(Width * Length + Height * Length + Height * Width)$

Experiment #3:

In this experiment, you need to define two packages named as "Pack1" and "Pack2" as follows:

Package 1 contains:

Class AccessDemo1: *public*

- Associate each access modifier to an integer variable

Class AccessDemo2: *default*

- Define a function that print the values of all fields defined in AccessDemo1
- What are your observations?

Package 2 contains:

Class AccessDemo3: default

- Define a function that print the values of all fields defined in AccessDemo1
- Force this function to print the value related to the protected integer without having compilation error.
- What are your observations?

Good Luck!

The University Of Jordan
Faculty of Engineering and Technology
Computer Engineering Department

Object-oriented Engineering Problem Solving Laboratory
Instructor: Dr. Khalid A. Darabkh

Lab 9: Exception Handling

Name: ……, **ID:** ……, **Section Number:** ……

Experiment #1:

- Write a Java program that enters an 8-digit string for a birth date using JOptionPane. The first two digits in the string are the month of birth, the next two are the day and the remaining four are the year. The Java program should squeeze out these substrings and display them on the screen. You need to make sure that what are inserted concerning the month, day, and year are not out of range and have a valid integer format.

Sample Output:

Suppose the string entered is: 08311978
Then prints:

month of birth: 08
day of birth: 31
year of birth: 1978

Suppose the string entered is: 01401979
Then prints:

Month 01 has only 30 days. Try again.

Suppose the string entered is: 0222197x
Then prints:

Invalid format for a year

Experiment #2:

Write a java class called *ExceptionDemo* that contains a declaration for an array of variable length and type *float* and two methods as follows:

- *Input* method which ensures inserting acceptable integer array length (i.e., integer and not less than zero) as well as elements of type *float*. This method should throw **DuplicateValueException** (needs to be defined as a checked exception) if the user inputs a value that already resides in the array. Furthermore, if an attempt is made to access an element outside the array bounds, you should catch the **ArrayIndexOutOfBoundsException** and display an appropriate error message.

- *Search* method which simply searches for a certain array element. Actually, it should throw a **NumberNotFoundException** (needs to be defined as an unchecked exception) if a particular element (value) cannot be found in the array during a search.

Note: You need to call aforementioned methods in the executable function properly. In all wrong cases, appropriate error messages should be displayed. The program should continue normal execution after handling the exceptions.

Good Luck!