

DSA Week 4 activities

This week, you are required to complete two questionnaires and one lab.

- a. In this print out, answer all week 4 questions.
- b. Also, in this print out, complete week 4 lab 1 & 2 using the lab computers.

Note: You can complete the activities in any order, however, make afford to complete and understand everything which prepares you for well for test 1, test 2, Major Assignment, Mid Semester Exam & Final Exam.

DSA Week 4 Questions

1. From the lecture, what are three principles of OODP?

Abstraction, encapsulation & modularity

2. What is abstraction?

The notion of **abstraction** is to distil a complicated system down to its most fundamental parts. Then describe the parts of a system which involves naming them and explaining their functionality.

Further, applying abstraction to the design of data structures give us abstract data types (ADT).

3. What is encapsulation?

Encapsulation is the principle that different components of a software system should not reveal the internal details of their respective implementations.

Encapsulation allows a programmer freedom to implement the details of a component, without concern that other programmers will be writing code that intricately depends on those internal decisions.

4. What is modularity?

Modularity refers to an organizing principle in which different components of a software system are divided into separate functional units.

Modern software systems typically consist of several different components that must interact correctly in order for the entire system to work properly. Keeping these interactions straight requires that these different components be well organized

5. What is polymorphism?

The word *polymorphism* literally means “many forms.”

In OOD, it refers to the ability of a reference variable to take different forms.

6. What is ADT?

ADT is a mathematical model of a data structure that specifies the type of data stored, the operations supported and the type of parameters of the operations. ADT can be expressed by an interface which are simply a list of method declarations.

7. Discuss method overriding and method overloading.

Method overriding is a method of superclass that is re-defined in the subclass. In other words, the subclass method over the superclass method when called in the program.

For example:

```
class s {  
    public void a(){  
    }  
}  
class t extends s {  
    Public void a(){  
    }  
}
```

Method overloading is a method name used by more than one method in the same class.

For example:

```
class T {  
    public int a(){  
    }  
    Public void a (int x){  
    }  
}
```

8. Explain the difference between throwing exceptions and catching exceptions.

The throw statement allows you to create a custom error. The throw statement is used together with an **exception type** defined by Java (examples ArithmeticException, FileNotFoundException, ArrayIndexOutOfBoundsException, SecurityException).

The try statement allows you to define a block of code to be tested for errors while it is being executed. The catch statement allows you to define a block of code to be executed if an error occurs in the try block.

9. Explain the difference between widening conversions and narrowing conversions.

https://www.tutorialspoint.com/java/java_type_casting.htm

Widening type casting is also known as **implicit type casting** in which a smaller type is converted into a larger type, it is done by the compiler automatically.

Here is the hierarchy of widening type casting in Java:

byte > short > char > int > long > float > double

The compiler plays a role in the type conversion instead of programmers. It changes the type of the variables at the compile time. Also, type conversion occurs from the small data type to large data type only.

Narrowing type casting is also known as **explicit type casting** or **explicit type conversion** which is done by the programmer manually. In the narrowing type casting a larger type can be converted into a smaller type.

When a programmer changes the variable type while writing the code. We can use the cast operator to change the type of the variable. For example, double to int or int to double.

Below is the syntax for narrowing type casting i.e., to manually type conversion:

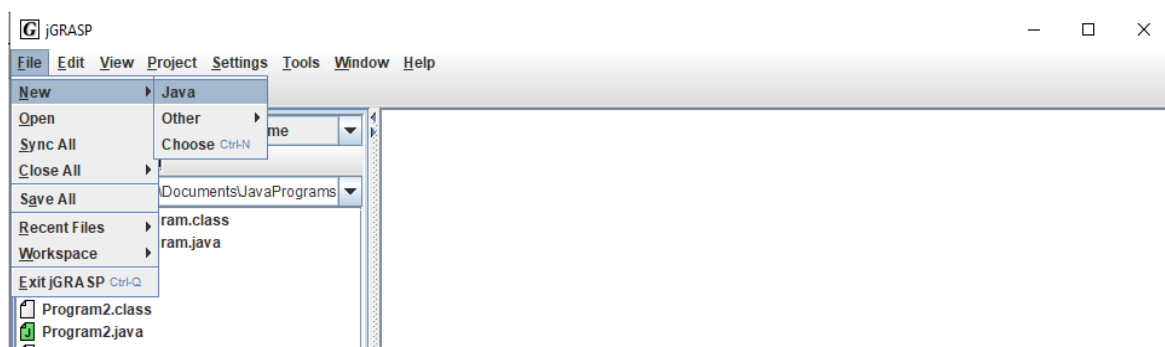
double doubleNum = (double) num;

The above code statement will convert the variable to double type.

DSA Week 4 Lab Activity (Week4Lab1)

Using the lab computers create the following Java program using jGrasp!

Step 1: Login to your lab computer and create a new java file in jGrasp.



Step 2: When the window below appears. Type the following code into jGrasp.

```

/* DSA Week 4 Lab 1 */

public class Week4Lab1 {

    //define variables
    int payJerome;
    int payOdilia;
    int payMakali;

    //create a constructor
    public Week4Lab1() {
        payJerome = 8;
        payOdilia = 10;
        payMakali = 12;
    }

    //create a method called myMethod with no parameter
    static void myMethod(){
        System.out.println("My first Java Method works!");
    }

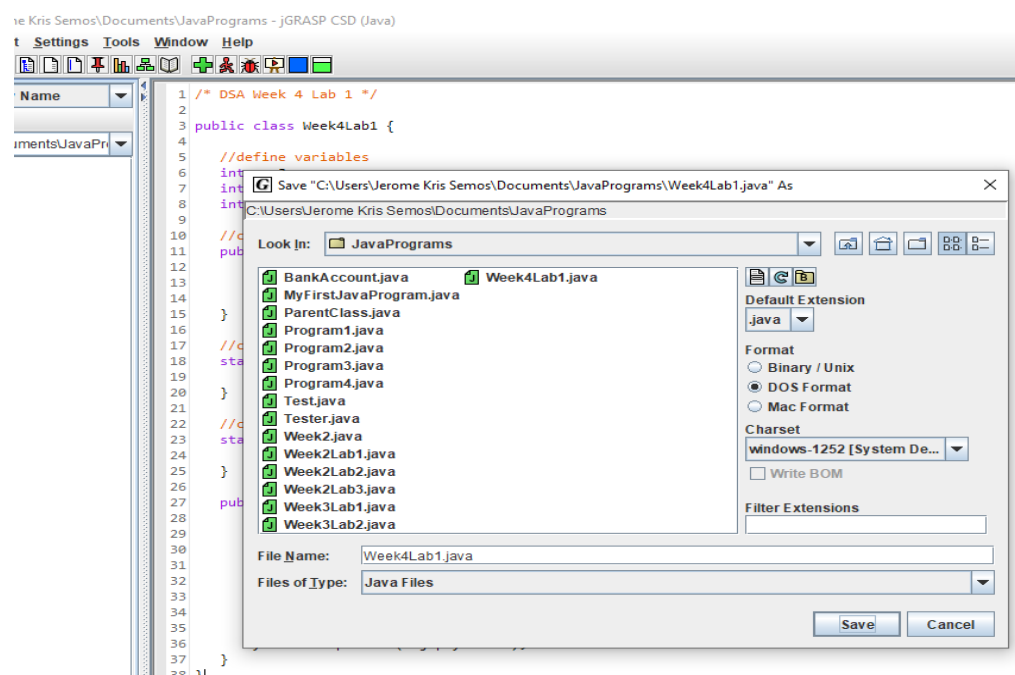
    //create a method called myMethod2 with parameter fname
    static void myMethod2(String fname){
        System.out.println(fname);
    }

    public static void main(String []args) {
        Week4Lab1 obj = new Week4Lab1();

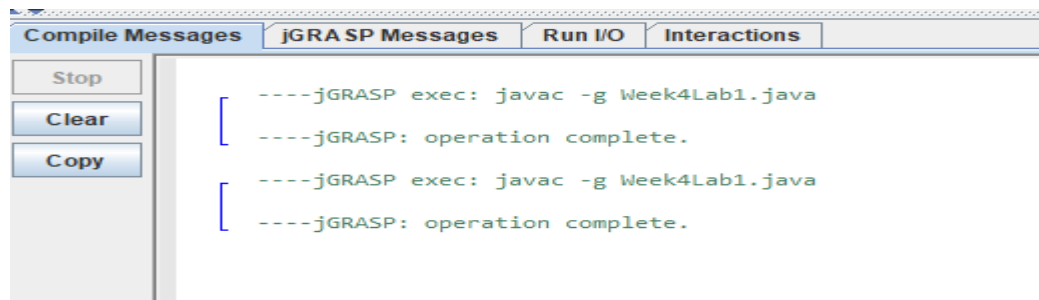
        myMethod();
        myMethod2("Jerome");
        System.out.println(obj.payJerome);
        myMethod2("Odilia");
        System.out.println(obj.payOdilia);
        myMethod2("Makali");
        System.out.println(obj.payMakali);
    }
}

```

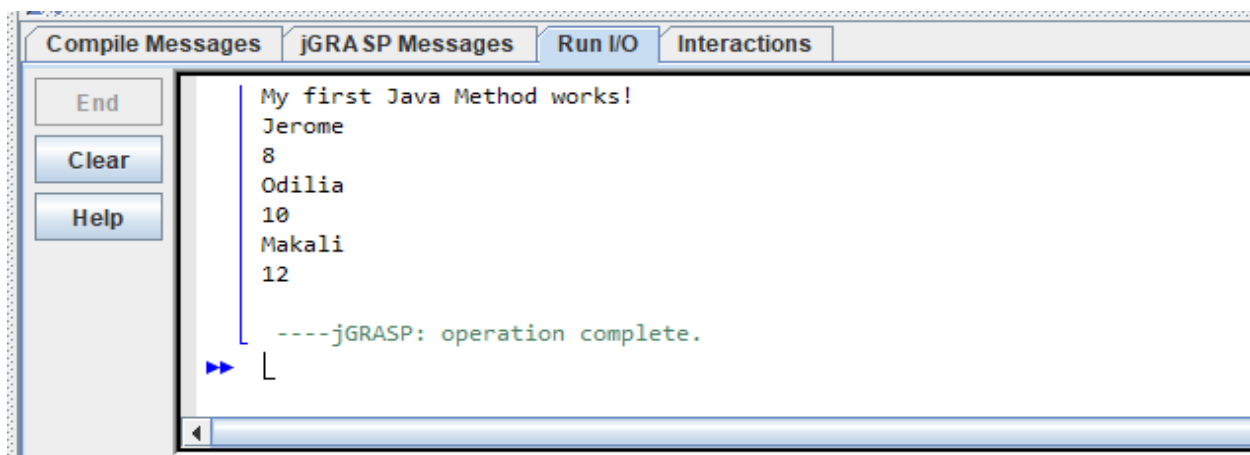
Step 3: Go to **file/save** to save your java program as **Week4Lab1**



Step 4: After saving, compile (**Go to Build/Compile**) to check for syntax errors. If compile is successfully then run (**Go the Build/Run**) your program. Below is example of the successful compiled program.



Step 5: Wait for the Java program to compile. If you see the message in the window below, it has successfully compiled.

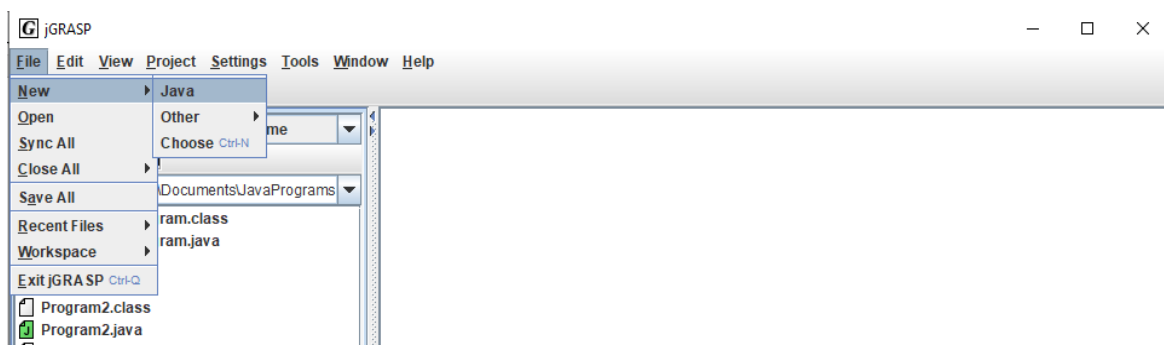


Step 8: Week4Lab1 Completed! Save your file for future Java lab activities.

DSA Week 4 Lab Activity (Week4Lab2)

Using the lab computers create the following Java program using jGrasp!

Step 1: Login to your lab computer and create a new java file in jGrasp.



Step 2: When the window below appears. Type the following code into jGrasp.

```

/* DSA Week 4 Lab 2 */

import java.util.Scanner; //import the Scanner class

public class Week4Lab2 {

    public static void main(String []args) {

        Scanner myObj = new Scanner(System.in); //create a Scanner object
        System.out.println("Enter your name");

        String name = myObj.nextLine(); //reads user input

        //Numbers input
        System.out.println("Enter your first number");
        int x = myObj.nextInt();
        System.out.println("Enter your second number");
        int y = myObj.nextInt();

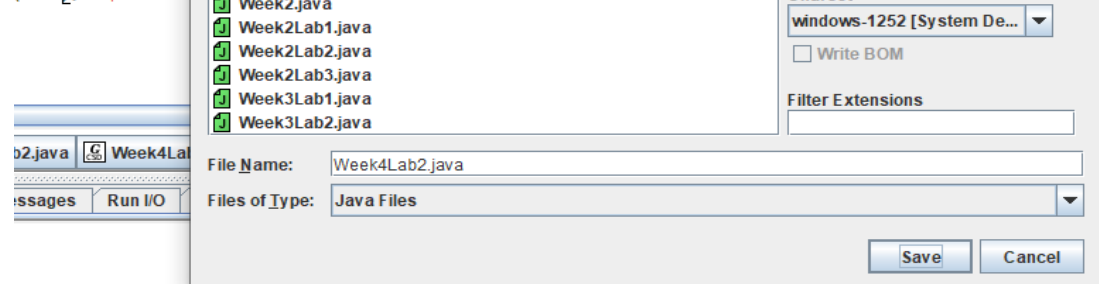
        int num = (x*y);
        int num2 = (x+y);
        int num3 = (x-y);
        int num4 = (x/y);

        System.out.println(name); // prints my name
        System.out.println(num); //prints multiplication
        System.out.println(num2); //prints additional
        System.out.println(num3); //prints subtraction
        System.out.println(num4); //prints division
    }
}

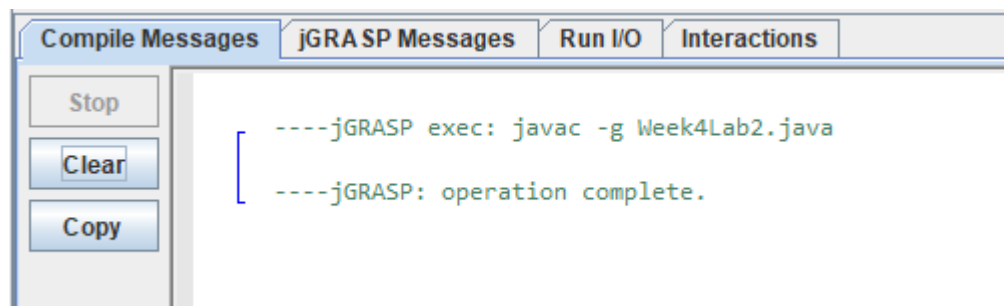
```

Step 3: Go to file/save to save your java program as Week4Lab2

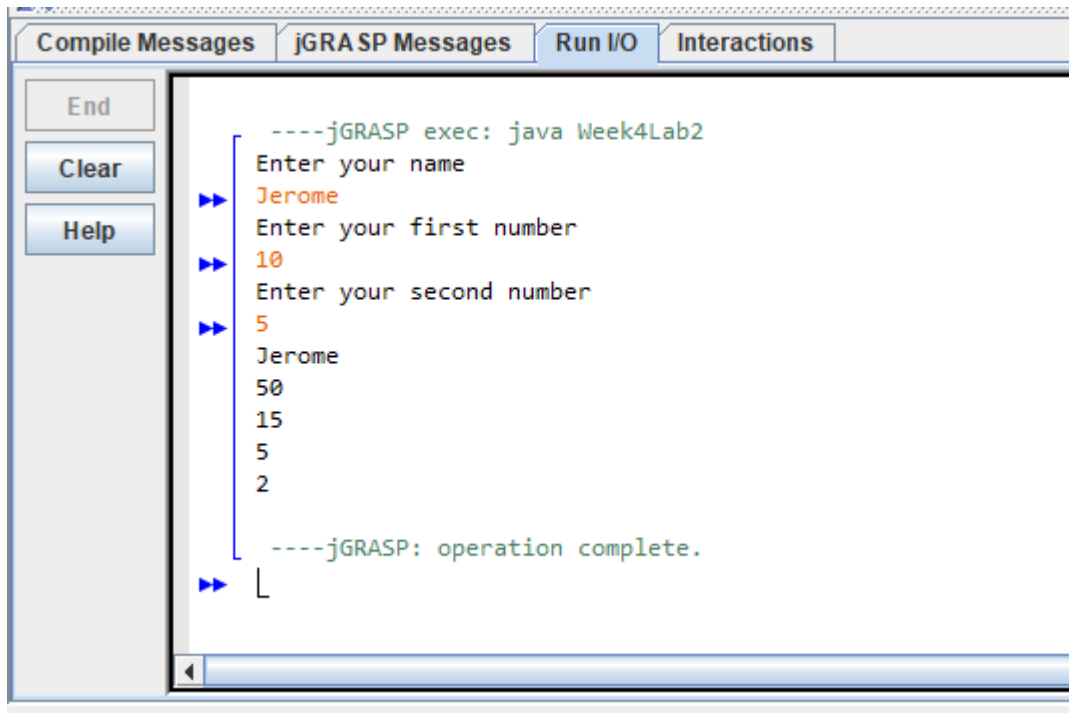
```
in(String []args) {  
  
    Scanner(System.in); //create a Scanner object  
    ("Enter your name");  
  
    j.nextLine(); //reads user input  
  
    ("Enter your first  
    Int();  
    ("Enter your second  
    Int();  
  
    (name); // prints  
    (num); //prints mu  
    (num2); //prints a  
    (num3); //prints s  
    (num4); //prints d
```



Step 4: After saving, compile (Go to Build/Compile) to check for syntax errors. Wait for the Java program to compile. If you see the message in the window below, it has successfully compiled.



Step 5: If compile is successfully then run (**Go the Build/Run**) your program. **Below is example of the successful compiled program.**



The screenshot shows the 'Run I/O' tab of the jGRASP IDE. On the left, there are buttons for 'End', 'Clear', and 'Help'. The main area displays the execution output of a Java program named 'Week4Lab2'. The output shows prompts for user input, followed by the user's responses: 'Jerome', '10', '5', 'Jerome', '50', '15', '5', and '2'. The program concludes with the message '----jGRASP: operation complete.'.

```
----jGRASP exec: java Week4Lab2
Enter your name
Jerome
Enter your first number
10
Enter your second number
5
Jerome
50
15
5
2
----jGRASP: operation complete.
```

- a) When the prompt appears type your name into the console and hit enter on your keyboard (refer to screenshot above for guidance)
- b) Next type a whole number (not decimals) and hit enter on the keyboard.
- c) Lastly, type another whole number and hit enter on the keyboard.

Step 6: If successful your program should display an output like shown in the screenshot above.

Step 7: Week4Lab2 Completed! Save your file for future Java lab activities.