

<b>NAME</b>	JACOB JOHN
<b>REGISTER NO.</b>	16BCE2205
<b>E-MAIL</b>	<a href="mailto:jacob.john2016@vitstudent.ac.in">jacob.john2016@vitstudent.ac.in</a>
<b>COURSE</b>	Java Programming

## **LAB ASSESSMENT #1**

### Jagged Arrays

If there are 4 batches in BTech - “CSE1007” course, read the count of the slow learners (who have scored <25) in each batch. Tutors should be assigned in the ratio of 1:4 (For every 4 slow learners, there should be one tutor). Determine the number of tutors for each batch. Create a 2-D jagged array with 4 rows to store the count of slow learners in the 4 batches. The number of columns in each row should be equal to the number of groups formed for that particular batch ( Eg., If there are 23 slow learners in a batch, then there should be 6 tutors and in the jagged array, the corresponding row should store 4, 4, 4, 4, 4, 3). Use for-each loop to traverse the array and print the details. Also print the number of batches in which all tutors have exactly 4 students.

### Code

```

/*
Jacob John
*/

package assignment2;
import java.util.Scanner;

public class multiDim {

    public static void main(String[] args) {
        int i, j;
        double t;

        // Declaring 2-D array with 4 rows
        int arr[][] = new int[4][];

        // input for each batch
        Scanner sc = new Scanner(System.in);
        for(i = 0; i < arr.length; i++){
            System.out.print("Enter number of students for batch " + (i+1) + ":");

            t = sc.nextDouble();
            arr[i] = new int[(int)Math.ceil(t/4)];
            for(j = 0; j < arr[i].length; j++)
            {
                if(t >= 4)

```

```

        arr[i][j] = 4;
    else
        arr[i][j] = (int)t;
    t = t - 4;
}
sc.close();

// Displaying the values of 2D Jagged array
int cfour = 0;
System.out.println("Contents of 2D Jagged Array");
for (i = 0; i < arr.length; i++) {
    for (j = 0; j < arr[i].length; j++) {
        System.out.print(arr[i][j] + " ");
        if(arr[i][j] == 4)
            cfour++;
    }
    System.out.println();
}

System.out.println("Number of tutors with 4 students are: " + cfour );
}
}

```

## Output

The screenshot shows the Eclipse IDE interface. The code editor displays the Java code for `multiDim.java`. The package explorer shows multiple projects like `Assignment_1`, `assignment2`, and `assignment3`. The console tab at the bottom shows the execution of the program, prompting for student counts and displaying the contents of the 2D jagged array and the count of tutors with 4 students.

```

Eclipse - assignment2/src/assignment2/multiDim.java - Eclipse
Java - JavaSE-1.8
File Edit Source Refactor Refactor Navigate Search Project Run Tools Window Help 60° 86% Sat Jan 12 1:13:33 AM
Quick Access
Package Explorer
Assignment_1 [javabyravindra master]
assignment2 [Java-Programming master]
JRE System Library [JavaSE-1.8]
src
assignment2
Bank.java
decryption.java
encryption.java
multiDim.java
StringHandle.java
assignment3 [Java-Programming master]
src
assignment3
TwinsPrime.java
primespackage
JRE System Library [Java SE 8 [1.8.0_162]]
Data-Types [javabyravindra master]
Scanner-Class [javabyravindra master]
testSpace [javabyravindra master]
Variables [javabyravindra master]
StringHandle.java multiDim.java Bank.java TwinsPrime.java
Task List
Find All Activate...
Outline
assignment2
multiDim
main(String[])
Javadoc Declaration Console Progress Azure Explorer Servers Cross References
<terminated> multiDim [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_162.jdk/Contents/Home/bin/java (Jan 12, 2019, 1:13:13 AM)
Enter number of students for batch 1: 27
Enter number of students for batch 2: 10
Enter number of students for batch 3: 23
Enter number of students for batch 4: 8
Contents of 2D Jagged Array
4 4 4 4 4 3
4 4 2
4 4 4 4 4 3
4 4
Number of tutors with 4 students are: 15
Building workspace: (100%)

```

## String Handling

Eg., Write a program to read a chemical equation and find out the count of the reactants and the products. Also display the count of the number of molecules of each reactant and product.

Eg., For the equation,  $2\text{NaOH} + \text{H}_2\text{SO}_4 \rightarrow \text{Na}_2\text{SO}_4 + 2\text{H}_2\text{O}$ , the O/P should be as follows.

- Reactants are 2 moles of NaOH, 1 mole of H<sub>2</sub>SO<sub>4</sub>.
- Products are 1 mole of Na<sub>2</sub>SO<sub>4</sub> and 2 moles of H<sub>2</sub>O.

Code

```
/*
Jacob John
*/

package assignment2;

import java.util.Scanner;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

class Coefficient{
    public static String extractCoefficient(String str) {
        String pattern = "^\d";

        // Create a Pattern object
        Pattern r = Pattern.compile(pattern);

        // Now create matcher object.
        Matcher m = r.matcher(str);
        if(m.find()){
            return m.group(0);
        }
        else{
            return "1";
        }
    }

    public static String removeCoefficient(String str) {
        String pattern = "[^\d](.*)";

        // Create a Pattern object
        Pattern r = Pattern.compile(pattern);

        // Now create matcher object.
        Matcher m = r.matcher(str);
        if (m.find()) {
            return m.group(0);
        }
        else {
            return str;
        }
    }
}
```

```

        }

    }

public class StringHandle {
    public static void main(String args[])
    {
        int i;
        //get input
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter equation: ");
        String eq = sc.nextLine();
        sc.close();

        //define re for splits
        String arrow = "(\\s->\\s)|(\\s->)|(->\\s)|(->)";

        // split into reactants and products
        String[] words = eq.split(arrow);
        String[] reactant = words[0].split("\\+");
        String[] product = words[1].split("\\+");

        String[] rmoles = new String[10];
        String[] pmoles = new String[10];

        for(i = 0; i < reactant.length; i++) {
            reactant[i] = reactant[i].replaceAll("\\s+", "");
            rmoles[i] = Coefficient.extractCoefficient(reactant[i]);
        }

        for (i = 0; i < product.length; i++) {
            product[i] = product[i].replaceAll("\\s+", " ");
            pmoles[i] = Coefficient.extractCoefficient(product[i]);
        }

        System.out.print("(1) Reactants are ");
        if(reactant.length > 1) {
            for(i = 0; i < reactant.length; i++)
            {
                if(i == 0)
                    System.out.print(rmoles[i] + " mole/s of " +
Coefficient.removeCoefficient(reactant[i]));
                else if(i == reactant.length-1)
                    System.out.print(" and " + rmoles[i] + " mole/s of " +
                        + Coefficient.removeCoefficient(reactant[i]));
                else
                    System.out.print(", " + rmoles[i] + " mole/s of " +
Coefficient.removeCoefficient(reactant[i]));
            }
        }
        else {
            System.out.print(rmoles[0] + " mole/s of "

```

```

        + Coefficient.removeCoefficient(reactant[0]));
    }

System.out.println();
System.out.print("(2) Products are ");
if (product.length > 1) {
    for (i = 0; i < product.length; i++) {
        if (i == 0)
            System.out.print(pmoles[i] + " mole/s of " +
Coefficient.removeCoefficient(product[i]));
        else if (i == product.length - 1)
            System.out.print(" and " + pmoles[i] + " mole/s of " +
Coefficient.removeCoefficient(product[i]));
        else
            System.out.print(", " + pmoles[i] + " mole/s of " +
Coefficient.removeCoefficient(product[i]));
    }
} else {
    System.out.print(pmoles[0] + " mole/s of " +
Coefficient.removeCoefficient(product[0]));
}
}
}
}

```

## Output

The screenshot shows the Eclipse IDE interface with the following components:

- Package Explorer:** Shows the project structure with three projects: Assignment\_1, assignment2, and assignment3.
- StringHandle.java:** The code editor displays the Java code for the StringHandle class. The code defines a static method extractCoefficient that uses regular expressions to find coefficients in a string. It also contains a main method that prints the reactants and products of a chemical equation.
- Task List:** A panel on the right showing various tasks and their status.
- Outline:** A panel showing the class hierarchy and member details.
- Console:** The bottom panel shows the terminal output of the program running. It prints the reactants (2 mole/s of NaOH and 1 mole/s of H2SO4) and the products (1 mole/s of Na2SO4 and 2 mole/s of H2O).

## Inheritance

Eg: Assume that a bank maintains two kinds of accounts for customers, one called as savings account and the other as current account. The savings account provides compound interest and withdrawal facilities but no cheque book facility. The current account provides cheque book facility but no interest. Current account holders should maintain a minimum balance and if the balance falls below this level, a service charge is imposed.

Create a class account that stores customer name, account number and type of account. From this derive the classes cur\_acct and sav\_acct to make them more specific to their requirements. Include necessary member functions in order to achieve the following tasks :

- Accept deposit from a customer and update the balance.
- Display the balance
- Compute and deposit interest.
- Permit withdrawal and update the balance.
- Check for the minimum balance, impose penalty, necessary, and update the balance.

### Code

```
/*
Jacob John
*/

package assignment2;

import java.util.Scanner;

class Account {
    int ano;
    float bal;

    public float getBal() {
        return bal;
    }

    public void setBal(float balance) {
        bal = balance;
    }
}

class savings extends Account {

    savings(int accnum) {
        ano = accnum;
        bal = 0;
        System.out.println("--Savings account created--");
        System.out.println("Acc. No.: " + ano);
        System.out.println("Balance: " + bal);
    }
}
```

```

        public static void compound(float rate, float time, float principle) {

            float compoundInterest = (float) (principle * Math.pow((1 + rate / 100),
time));

            System.out.println("The Compound Interest is : " + compoundInterest);
        }
    }

class current extends Account{

    float min;

    current(int accnum, float amt){
        ano = accnum;
        bal = 0;
        min = amt;
        System.out.println("--Current account created--");
        System.out.println("Acc. No.: " + ano);
        System.out.println("Balance: " + bal + " (Please add balance)");
        System.out.println("Minimum Balance: " + min);
    }

    public void withdraw(float amt) {
        if (bal < min)
            System.out.println("Below minimum balance!");
        else {
            bal = bal - amt;
            System.out.println("New Balance: " + this.getBal());
        }
    }
}

public class Bank {
    public static void main(String args[])
    {
        Scanner sc = new Scanner(System.in);

        savings s1 = new savings(1);
        System.out.print("Enter Amt: ");
        float amt = sc.nextFloat();
        s1.setBal(amt);
        System.out.println("New Balance: " + s1.getBal());

        System.out.println("\n" + "--Calculating Compound Interest--");
        System.out.print("Enter principle: ");
        float principle = sc.nextFloat();
        System.out.print("Enter rate: ");
        float rate = sc.nextFloat();
        System.out.print("Enter time: ");
    }
}

```

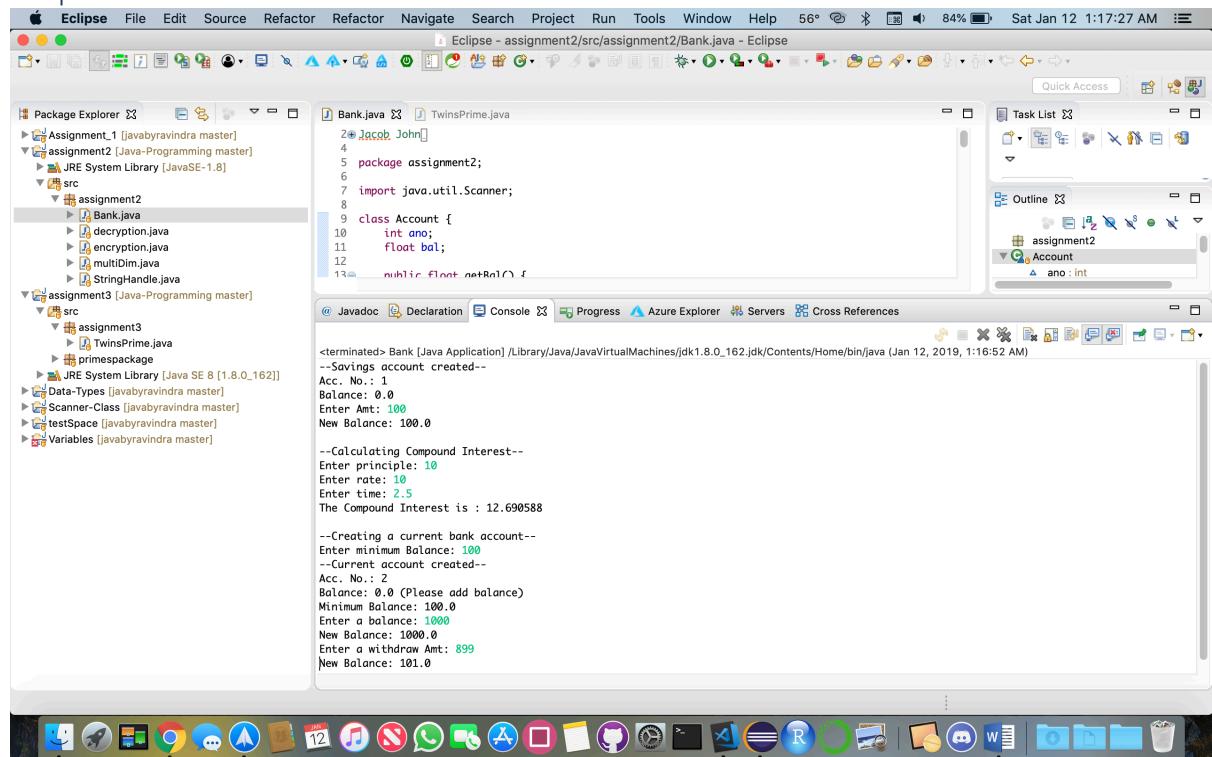
```
float time = sc.nextFloat();
savings.compound(rate, time, principle);

System.out.println("\n" + "--Creating a current bank account--");
System.out.print("Enter minimum Balance: ");
amt = sc.nextFloat();
current c1 = new current(2, amt);
System.out.print("Enter a balance: ");
float b = sc.nextFloat();
c1.setBal(b);
System.out.println("New Balance: " + c1.getBal());
System.out.print("Enter a withdraw Amt: ");
amt = sc.nextFloat();
c1.withdraw(amt);

sc.close();
}

}
```

## Output



## Packages and Sub packages

Write a program to demonstrate the knowledge of students in working with user-defined packages and sub-packages.

Eg., Within the package named ‘primespackage’, define a class Primes which includes a method checkForPrime() for checking if the given number is prime or not. Define another class named TwinPrimes outside of this package which will display all the pairs of prime numbers whose difference is 2. (Eg, within the range 1 to 10, all possible twin prime numbers are (3,5), (5,7)). The TwinPrimes class should make use of the checkForPrime() method in the Primes class.

Code

Prime.java

```
/*
 * Jacob John
 */

package primespackage;

public class Prime {

    public static boolean checkPrime(int num) {
        int temp;
        boolean isPrime = true;

        for (int i = 2; i <= num / 2; i++) {
            temp = num % i;
            if (temp == 0) {
                isPrime = false;
                break;
            }
        }

        return isPrime;
    }
}
```

TwinsPrime.java

```
/*
 * Jacob John
 */

package assignment3;

import java.util.Scanner;
import primespackage.Prime;

public class TwinsPrime {
```

```

public static void main(String args[]) {
    int i;

    //take input
    Scanner sc = new Scanner(System.in);
    System.out.print("Enter first number: ");
    int n1 = sc.nextInt();
    System.out.print("Enter second number: ");
    int n2 = sc.nextInt();
    sc.close();

    System.out.println("Twin prime numbers are: ");
    //checking twins prime
    for(i = n1; i <= n2; i++) {
        if(Prime.checkPrime(i) & Prime.checkPrime(i+2) & i > 1)
        {
            System.out.println("(" + i + "," + (i+2) + ")");
        }
    }
}

```

## Output

The screenshot shows the Eclipse IDE interface with the following components:

- Package Explorer:** Shows the project structure with packages like Assignment\_1, Assignment2, and assignment3.
- Code Editor:** Displays the `TwinsPrime.java` file containing the Java code for finding twin prime numbers.
- Task List:** Shows a search bar and a list of tasks.
- Outline:** Shows the class structure and the `main` method.
- Console:** Displays the terminal output of the program execution.

**Code in TwinsPrime.java:**

```

2* * Jacob John
4
5 package assignment3;
6
7import java.util.Scanner;
8
9 public class TwinsPrime {
10     public static void main(String args[]) {
11         int i;
12
13         //take input
14         Scanner sc = new Scanner(System.in);
15         System.out.print("Enter first number: ");
16         int n1 = sc.nextInt();
17         System.out.print("Enter second number: ");
18         int n2 = sc.nextInt();
19         sc.close();
20
21         System.out.println("Twin prime numbers are: ");
22         //checking twins prime
23         for(i = n1; i <= n2; i++) {
24             if(Prime.checkPrime(i) & Prime.checkPrime(i+2) & i > 1)
25             {
26                 System.out.println("(" + i + "," + (i+2) + ")");
27             }
28         }
29     }
30 }

```

**Console Output:**

```

<terminated> TwinsPrime [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_162.jdk/Contents/Home/bin/java (Jan 12, 2019, 1:19:45 AM)
Enter first number: 1
Enter second number: 10
Twin prime numbers are:
(3,5)
(5,7)

```