



2023-24

جب کوئی قوم فن اور  
علم سے عاری ہو  
جاتی ہے تو وہ غربت  
کو دعوت دیتی ہے  
اور جب غربت آتی  
ہے تو وہ ہزاروں  
جرائم کو جنم دیتی  
ہے۔

“When a nation  
becomes devoid  
of art and  
learning, it  
invites poverty  
and when  
poverty comes it  
brings in its  
wake thousands  
of crimes.”

-Sir Syed Ahmad  
Khan

CAMS2P01

Laboratory Course-II



MCA II Semester Lab Manual

DEPARTMENT OF COMPUTER SCIENCE

ALIGARH MUSLIM UNIVERSITY, ALIGARH

2023-2024

## Credits

The following lab manual up-gradation committee updated the lab manual:

- **Prof. Aasim Zafar (Chairperson)**
- **Prof. Arman Rasool Faridi**
- **Prof. Suhel Mustajab**
- **Dr. Faisal Anwar**
- **Dr. Mohammad Nadeem**

The following committee members originally designed the lab manual:

- **Prof. Mohammad Ubaidullah Bokhari**
- **Prof. Arman Rasool Faridi**
- **Dr. Faisal Anwer**
- **Prof. Aasim Zafar (Converner)**

**Design & Compilation:**

- **Dr. Faraz Masood**

**Revised Edition: January, 2024**

**Department of Computer Science,  
A.M.U.,  
Aligarh (U.P.) India**

**COURSE TITLE:** Laboratory Course-II  
**CREDIT:** 04  
**CONTINUOUS ASSESSMENT:** 40

**COURSE CODE:** CAMS2P01  
**PERIODS PER WEEK:** 06  
**EXAMS:** 60

## COURSE DESCRIPTION

This course is designed to help students in learning JAVA using object-oriented paradigm. Approach in this course is to take JAVA as a language that is used as a primary tool in many different areas of programming work.

## COURSE CONTENT

This course is designed to provide the students the opportunity to learn the differences between C++ and JAVA programming and to develop, debug, and execute JAVA programs.

## OBJECTIVES

This course is designed to help students in:

- ☐ Gaining knowledge about basic Java language syntax and semantics to write Java programs and use concepts such as variables, conditional and iterative execution methods etc.
- ☐ Understanding the fundamentals of object-oriented programming in Java, including defining classes, objects, invoking methods, etc., and exception handling mechanisms.
- ☐ Learning the use of arrays, access protection, wrapper classes etc.
- ☐ Understanding the principles of inheritance, packages, and interfaces.
- ☐ Learning to create applet and application and event handling, AWT controls, able to create GUI (frame, menu, button, text boxes, layout manager).
- ☐ Familiarizing with the important topics and principles of software development.

- ❑ Learning the ability to write a computer program to solve specified problems.
- ❑ Understanding to use the Java SDK environment to create, debug and run simple Java programs.

## OUTCOMES

After completing this course, the students would be able to:

- ❑ Understand the fundamental features of an object-oriented language like JAVA: object classes and interfaces, exceptions, and libraries of object collections.
- ❑ Understand how to implement, compile, test, and run JAVA programs comprising more than one class to address a particular software problem.
- ❑ Understand how to include arithmetic operators and constants in a JAVA program and able to deploy the JAVA applet and application program.
- ❑ Understand the concept of multithreading, multitasking, and multiprogramming.
- ❑ Understand the use of members of classes found in the JAVA API (such as the Math class, Wrapper classes).
- ❑ Understand the implementation of the keyboard/mouse events.
- ❑ Understand the concept of object-oriented programming, inheritance, constructors, interfaces, and packages.
- ❑ Understand the concept of Exception Handling, Files and Streams, Applets and Graphics, and also learn the concept of Applet classes, Applet life cycle and JAVA Swing (introductory part).
- ❑ How to take the statement of a business problem and, from this, determine suitable logic for solving the problem; then, be able to code that logic as a program written in JAVA.

# RULES AND REGULATIONS

Students are required to strictly adhere to the following rules.

- ❑ The students must complete the weekly activities/assignments well in time (i.e., within the same week).
- ❑ The students must maintain the Lab File of their completed activities/assignments in the prescribed format (**Appendix-1**).
- ❑ The students must get the completed weekly activities/assignments checked and signed by the concerned teachers in the lab in the following week. Failing which activities/assignments for that week will be treated as incomplete.
- ❑ At least **TEN (10)** such timely completed and duly signed weekly activities/assignments are compulsory, failing which students will not be allowed to appear in the final Lab Examination.
- ❑ The students need to submit the following three deliverables for each exercise duly signed by the Teacher:
  - ❖ Coding
  - ❖ Input /Output
- ❑ The students need to ensure that each question is assessed and signed by the teacher within the week/time.
- ❑ Late submissions would not be accepted after the due date.
- ❑ Cooperate, collaborate, and explore for the best individual learning outcomes, but copying is strictly prohibited.



## APPENDIX-1

### Template for the Index of Lab File

WEEK NO.	PROBLEMS WITH DESCRIPTION		PAGE NO.	SIGNATURE OF THE TEACHER WITH DATE
1	1#			
	2#			
	3#			
2	1#			
	2#			
	3#			
3	1#			
	2#			
	3#			

*Note: The students should use Header and Footer mentioning their roll no. & name in footer and page no in header.*

# WEEK #1

## OBJECTIVES

- ☐ To help the students understand the importance of programming in Object Oriented environment using JAVA
- ☐ To help the students in understanding the basic structure of JAVA Program
- ☐ To help students get familiar with the JAVA programming environment and IDE.
- ☐ To learn the students in writing, debugging and executing JAVA programs.
- ☐ How to read input from the keyboard?
- ☐ To learn the constant and variables.

## OUTCOMES

After completing this, the students would be able to:

- ☐ Write, debug, and execute simple JAVA programs.
- ☐ Use the constant and variables in JAVA.

## PROBLEMS

- 1# Write a Java program to print '*Hello World*'.
- 2# Write a java program to print the corresponding address of a student.
- 3# Write a Java program to calculate sum of two numbers.
- 4# Write a Java program to convert the given temperature in Fahrenheit to Celsius using the following conversion formula  $C = (F - 32) / 1.8$  and display the value in a tabular form.

## WEEK #2

### OBJECTIVE

- ☐ To learn the concepts of operators used in JAVA.
- ☐ To learn the various mathematics library in Java.

### OUTCOMES

After completing this, the students would be able to:

- ☐ use the operators in JAVA.
- ☐ use the various mathematic library functions.

### PROBLEMS

- 1# Write a java program for finding the sum, difference, product, quotient, minimum and maximum of any two integers.
- 2# Write a java program 'MyNumber.java' that performs following operations on a variable 'num' of type double:
  - i) Finds the round value of 'num' and stores the result in a variable numRound of type double.
  - ii) Finds the ceil value of 'num' and stores the result in a variable numCeil of type double.



**iii)** Finds the floor value of 'num' and stores the result in a variable numFloor of type double.

Cast 'num' to type int and stores the result in a variable numInteger of type int.  
Display output of numRound, numCeil, numFloor and numInteger on screen.

*Note:* Test your program with following value of num

num=56.764num=36.432

**3#** Write Java program to show uses of all Math class methods.

# WEEK #3

## OBJECTIVE

- ☐ To learn using JAVA programming coding: data types, variable, constants, operators, Control Statement (*if, switch, loops*)
- ☐ To learn break, continue statements, ternary operator, bit-wise operators, user-defined data types in JAVA, order of evaluation of different operators in Java.
- ☐ To learn the controls statements and loops.

## OUTCOMES

After completing this,

- ☐ The students would be able to use different control statements and loops available in JAVA.

## PROBLEMS

- 1# Write a java program to prints the count of odd and even no's entered.
- 2# Write a java program to print the squares and cubes for the numbers 1 to 5.
- 3# Write a java program that computes the sum of the reciprocals:

$$\frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} \dots \dots \dots + \frac{1}{10}$$

- 4# Write Java program to compute the sum of the 2+4+6+-----N Terms.

- 5# Shown below is a Floyd's triangle:

```
1
2 3
4 5 6
7 8 9 10
```

```
11 12 13 14
15 16 17 18 19
```

- i) Write a program to print the above triangle.
- ii) Modify the program to produce the following form of Floyd's triangle.

```
1
0 1
1 0 1
0 1 0 1
1 0 1 0 1
0 1 0 1 0 1
```

# WEEK #4

## OBJECTIVE

- ☐ To learn using JAVA programming coding: data types, variable, constants, operators, Control Statement (*if, switch, loops*)
- ☐ To learn break, continue statements, ternary operator, bit-wise operators, user-defined data types in JAVA, order of evaluation of different operators in Java.
- ☐ To learn the controls statements and loops.

## OUTCOMES

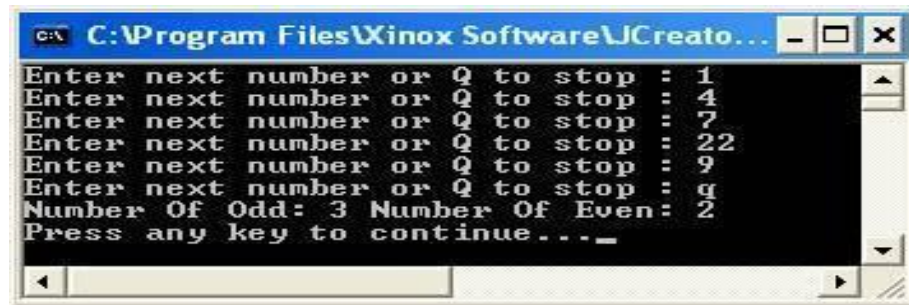
After completing this,

- ☐ The students would be able to use different control statements and loops available in JAVA.

## PROBLEMS

- 1# Write a program in the sequence 0, 1, 1, 2, 3, 5, 8, 13, 21 ..... is called *Fibonacci numbers*. Write a program using a *do...while* loop to calculate and print the first m *Fibonacci numbers*.
- 2# Write a program to accept three digits (i.e. 0 - 9) and print all its possible combinations. (*For example if the three digits are 1, 2, 3 than all possible combinations are: 123, 132, 213, 231, 312, 321*)
- 3# Write a Java Program which prompts the user to enter 4 numbers. The program will then computes and display their sum and their product.

- 4# Write a Java program which reads a 4-digit number and prints the digits on separate lines. (Each digit is printed on one line). Output of your program must be in the following format:



```
Enter next number or Q to stop : 1
Enter next number or Q to stop : 4
Enter next number or Q to stop : 7
Enter next number or Q to stop : 22
Enter next number or Q to stop : 9
Enter next number or Q to stop : q
Number Of Odd: 3 Number Of Even: 2
Press any key to continue...
```

- 5# The intersection method computes the intersection of two rectangles- that is, the rectangle that is formed by two overlapping rectangles: You call this method as follows: **Rectangle r3=r1.intersection (r2);**

Write a program that constructs two rectangle objects, prints them, and then prints their intersection. What happens when the rectangles do not overlap?

# WEEK #5

## OBJECTIVES

- ☐ To learn the object oriented features (classification, encapsulation, inheritance, polymorphism, abstraction) and Java features like secure, platform independent, portable, threading.
- ☐ To learn the concept of class like how to declare a class, how to define methods inside the class, how to access the properties of base class into derived class, Code reusability.
- ☐ To learn the use of arrays, access protection, wrapper classes.
- ☐ To learn the use of this keyword, use of toString ( ) method.

## OUTCOME

After completing this, the students would be able to:

- ☐ employ various types of selection constructs in a JAVA program
- ☐ demonstrate the hierarchy of JAVA classes to provide a solution to a given set of requirements.

## PROBLEMS

1# Write Java program involving two classes: *OddAndEven* & *TestOddAndEven*.

*OddAndEven* has the following:

- Instance variables *countOfOdd* and *countOfEven* both of type *int*.
- A method *addNumber* that takes a number as parameter and increment *countOfOdd*, if the number is odd, else increment *countOfEven*.
- A method *toString* that returns a string in the form: "Number of Odd: x, Number of Even: y", where x and y are the values of the instance variables.



The *TestOddAndEven* class first creates *OddAndEven* object, then in a loop, read a number and use it to call the *addNumber* method until the user enters Q. Finally, it prints the count of odd and even numbers entered.

**2#** Design a class *Circle* and implement the following methods:

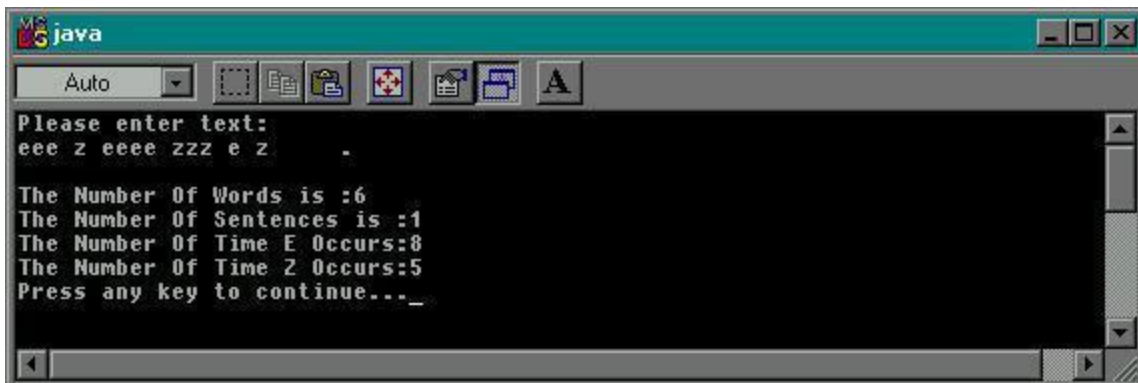
- Define a circle method to compute its area
- Define a circle method to compute its perimeter
- Define a method that takes a given point represented by a pair of numbers and checks whether or not the point is inside the circle.

The circle class needs to have instance variables to store the radius of the circle, and the x and y coordinates of the center. Add main program to test the class *Circle* repeatedly, until user enters negative value for the radius of the circle.

**3#** Write a program in Java that reads in text and prints as output the following:

- The number of words in the text
- The number of sentences in the text
- The number of times the letter "e" occurs in the text
- The number of times the letter "z" occurs in the text

(Note: Use *StringTokenizer* class)



```
java
Auto
Please enter text:
eee z eeee zzz e z
The Number Of Words is :6
The Number Of Sentences is :1
The Number Of Time E Occurs:8
The Number Of Time Z Occurs:5
Press any key to continue...
```

**4#** A sales person is paid commission based on the sales he makes as shown by the following table:

Sales	Commission
Under ₹500	2% of Sales
Between ₹500 to ₹5000	5% of Sales
₹5000 and Above	8% of Sales

Write a class *Commission* which has an instance variable *sale*, an appropriate constructor, and a method *commission ()* that returns the commission.

Now write a demo class to test the *Commission* class by reading a sale from the user, using it to create a *Commission* object after validating that the value is not negative. Finally, call the *commission ()* method to get and print the commission. If the sales are negative, your demo should print the message "Invalid Input".

## WEEK #6

## OBJECTIVES

- ☐ To learn the object oriented features (classification, encapsulation, inheritance, polymorphism, abstraction) and Java features like secure, platform independent, portable, threading.
- ☐ To learn the concept of class like how to declare a class, how to define methods inside the class, how to access the properties of base class into derived class, Code reusability.
- ☐ To learn the use of arrays, access protection, wrapper classes.
- ☐ To learn the use of this keyword, use of toString ( ) method.

## OUTCOME

After completing this, the students would be able to:

- ☐ employ various types of selection constructs in a JAVA program
- ☐ demonstrate the hierarchy of JAVA classes to provide a solution to a given set of requirements.

## PROBLEMS

1# The certain instructor assigns letter grade for his course based on the following table:

Score	Grade
>=90	A+
>=85	A
>=80	B+
>=75	B
>=65	C+
>=60	C
>=55	D+

$\geq 50$	D
$< 50$	F

Write a class, Grader, which has an instance variable, score, an appropriate constructor and appropriate method *letterGrade()* that returns the letter grade as a String.

Now write a demo class to test the Grader class by reading a score from the user, using it to create a Grader object after validating that the value is not negative and is not greater than 100. Finally, call the *letterGrade()* method to get and print the grade.

**2#** Implement a Student class with the following fields, constructors and methods:

**Fields:**

- *name;*
- *totalScore;*
- *numberOfQuizzes;*

**Constructors:**

- *public Student(String name, double score)*
- *public Student(double score, String name)*
- *public Student(String name)*

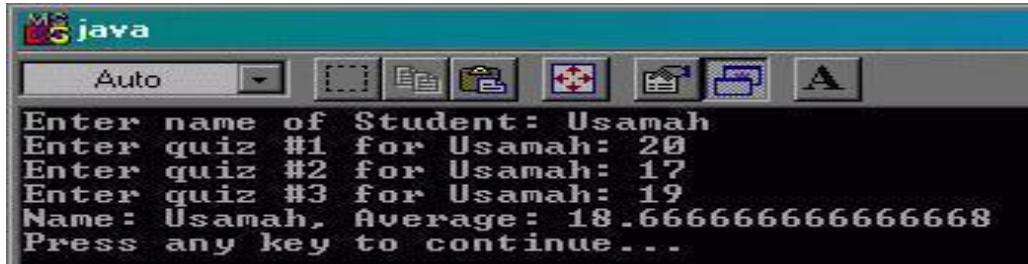
**Methods:**

- *public String getName()*
- *public double getAverage()* //this should return zero if no quiz has been taken.
- *public double getTotalScore()*
- *public void addQuiz(double score)*
- *public void printStudent()* //this should print the student's name and average score.

- *public String toString()*

Write an application TestStudent that reads a student name and use the Student class to create a Student object. Then read the scores of the student in three quizzes and add each to the totalScore of the student using addQuiz() method and print the student object.

( Note: Make use of this key word wherever it can be used ).



```
Enter name of Student: Usamah
Enter quiz #1 for Usamah: 20
Enter quiz #2 for Usamah: 17
Enter quiz #3 for Usamah: 19
Name: Usamah, Average: 18.666666666666668
Press any key to continue...
```

**3#** Write a program to design a class to represent a bank account. Include the following members.

**Date members:**

- Name of depositor
- Account Number
- Type of account
- Balance account in the account

**Methods:**

- To assign initial values
- To deposit an account
- To withdraw an account after checking balance.
- To display the name and balance

# WEEK #7

## OBJECTIVES

- ☐ To learn the concept Interface, how to implement it,
- ☐ To learn the use of member functions and how we access them, using interfaces for code reusing,
- ☐ To learn converting between class and interface types, using interfaces for callbacks; Polymorphism, Inheritance

## OUTCOMES

After completing this, the students would be able to:

- ☐ handle interfaces, converting between class and interface types, callbacks, Polymorphism, Inheritance, Inheriting instance fields and methods.
- ☐ access the member functions of a class.

## PROBLEMS

- 1# Write a program that reads in a sentence from the user and prints it out with each word reversed, but with the words and punctuation in the original order:



- 2# Write a program where interface can be used to support multiple inheritances. Develop a standalone Java program for this.
- 3# Write a program that reads in three strings and sorts them lexicographically.  
*Hint:* Enter strings: Charlie Able Banker



Output: Able Banker Charlie

- 4#** Implement the classes for the shapes using an interface for the common methods, rather than inheritance from the super class, while still Shape as a base class.

# WEEK #8

## OUTCOMES OBJECTIVES

- ☐ To learn the toString method in Java.
- ☐ To learn the use of member functions and how we access them.,
- ☐ To learn the concept and usage of constructor in Java.

After completing this, the students would be able to:

- ☐ understand the use of toString method in Java.
- ☐ understand the usage of constructor in class.
- ☐ access the member functions of a class.

## PROBLEMS

- 1# Implement a super class Person. Make two classes, Student and Instructor, inherit from Person. A person has a name and a year of birth. A student has a major, and an instructor has a salary. Write the class definitions, the constructors, and the methods toString for all classes. Supply a test program that tests these classes and methods.
- 2# Define a class Employee having private members – id, name, department, salary. Define default and parameterized constructors. Create a subclass called "Manager" with private member bonus. Define methods accept and display in both the classes. Create n objects of the Manager class and display the details of the manager having the maximum total salary (salary+bonus).
- 3# Write a Java program to create a super class Vehicle having members Company and price. Derive 2 different classes *LightMotorVehicle* (members – mileage)

and *HeavyMotorVehicle* (members – capacity-in-tons). Accept the information for n vehicles and display the information in appropriate form. While taking data, ask the user about the type of vehicle first.

# WEEK #9

## OBJECTIVES

- ☐ To learn the concept of Access control: private access, public access, protected access and package access.
- ☐ To learn the use of member functions and how we access them.
- ☐ To learn the concept and usage of constructor in Java.

## OUTCOMES

After completing this, the students would be able to:

- ☐ understand the concept of Access control, its purpose, its characteristics, how many types of Access control.
- ☐ understand the usage of constructor in class.
- ☐ access the member functions of a class.

## PROBLEMS

- 1# A bank maintains two kinds of accounts - Savings Account and Current Account. The savings account provides compound interest, deposit and withdrawal facilities. The current account only provides deposit and withdrawal facilities. Current account holders should also maintain a minimum balance. If 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 Curr-acct and Sav-acct. Include the necessary methods in order to achieve the following tasks:

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

**2#** Define a class called fruit with the following attributes:

- Name of the fruit
- Single fruit or bunch fruit

Define a suitable constructor and *displayFruit()* method that displays values of all the attributes. Write a program that creates 2 objects of fruit class and display their attributes.

**3#** Write a program where an interface can be used to support multiple inheritances. Develop a standalone Java program for this.

**4#** Implement the classes for the shapes using an interface for the common methods, rather than inheritance from the superclass, while still Shape as a base class.

**5#** Create a package called "Arithmetic" that contains methods to deal with all arithmetic operations. Also, write a program to use the package.

**6#** Write a program to make use of a parameterized method inside a class. Take the following case: Create a class Box and define a method in this class that will return the volume of the box. Initialize two objects for your class and print out the volumes, respectively.

# WEEK #10

## OBJECTIVES

- ☐ To learn the concept of Exception handling.
- ☐ To learn the importance of exceptions.
- ☐ To learn the concept and way of throwing exceptions.

## OUTCOMES

After completing this, the students would be able to:

- ☐ Understand the features of Exception handling.
- ☐ Understand the concept of Exception handling.
- ☐ Use throwing exceptions.

## PROBLEMS

- 1# Write a program that calls a method that throws an exception of type *ArithmeticException* at a random iteration in a *for* loop. Catch the exception in the method and pass the iteration count to the calling method when the exception occurred by using an object of an exception class you define.
- 2# Write a program that will count the number of characters in a file
- 3# In a small firm, employee numbers are given in serial numerical order, that is 1, 2, 3 etc. Write a menu-driven program to perform the following operations:



- i) Create a file of employee data with the following information: Employee No, Name, Sex, Gross Salary.
  - ii) Append the data of a new employee joining the firm.
  - iii) If a given employee leaves, delete his record.
  - iv) If the gross salary of a given employee increases, update the gross salary.
  - v) Display the record of :
    - a. a given employee or
    - b. all employees.
- 4#** Write a program to create a sequential file that could store details about five products. Details include product code, cost, and number of items available and are provided through the keyboard.

# WEEK #11

## OBJECTIVES

- ☐ To learn the file handling using Java.
- ☐ To learn various operations performed in file using file handling concepts.

## OUTCOMES

After completing this, the students would be able to:

- ☐ Work with Files and Streams.

## PROBLEMS

- 1# Write a Java program which reads student grades from a text file called grades.txt and prints only the corresponding letter grades into a file called letter.txt. The letter grades are assigned according to the following table. Assume that the grades.txt file can have any number of students' grades. Hint: The last number in the grades.txt file is -1

Score	Grade
$\geq 90$	A+
$\geq 85$	A
$\geq 80$	B+

$\geq 75$	B
$\geq 65$	C+
$\geq 60$	C
$\geq 55$	D+
$\geq 50$	D
$< 50$	F

- 2#** Write a program to read a, b, c from data file and store the roots of the quadratic equation in the output file. You must open your output file in append mode.
- 3#** Develop an applet that receives three numeric values as input from the user and then displays the largest of the three on the screen. Write HTML pages and test the applet.

# WEEK #12

## OBJECTIVES

- ❑ To learn the concept of applet and their applications.

## OUTCOMES

After completing this, the students would be able to:

- ❑ Understand applets in terms of drawing graphical shapes, colours, fonts, drawing complex shapes, reading text input inside applet.

## PROBLEMS

1# Write applets to draw the following shapes:

a) Cone    b) Cylinder    c) Square inside a circle

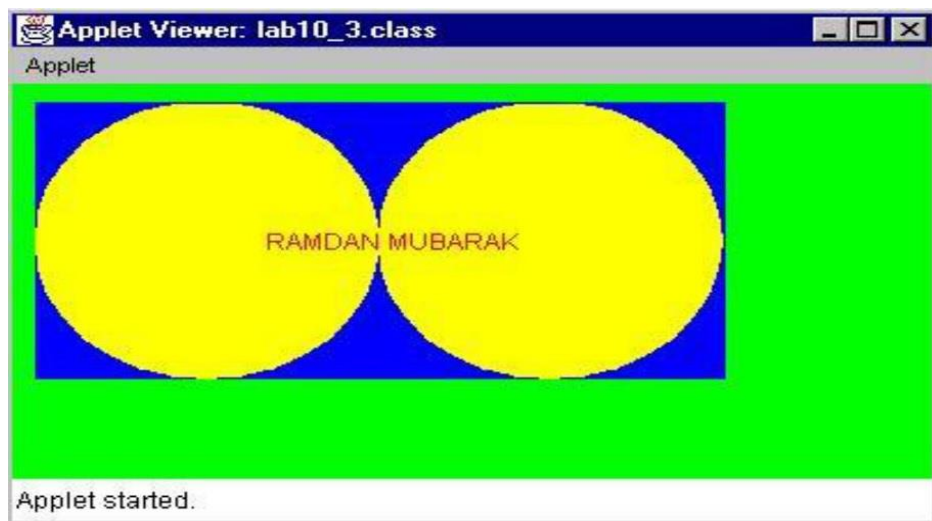
2# Write an applet that will display the following on a green background. Use the following dimension:

**Rectangle** : (10, 10, 300, 150), Fill colour: blue

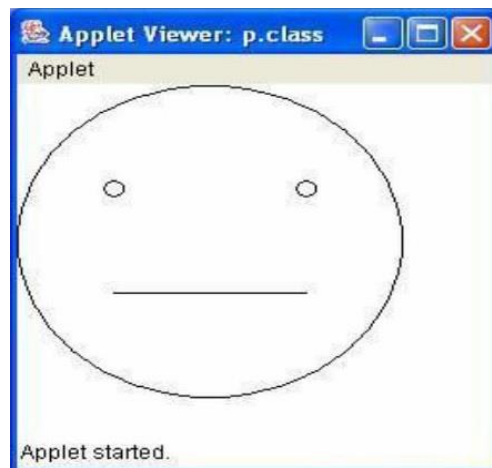
**Left Circle** : (10, 10, 50, 150), Fill Colour: Yellow

**Right Circle**: (159, 10, 150, 150), Fill colour: Yellow

**Text** : (110, 90), colour: Red



3# Write a JAVA Applet program to plot the following face:



# WEEK #13 AND #14

## OBJECTIVES

- ❑ To learn the concept of applet and their applications.

## OUTCOMES

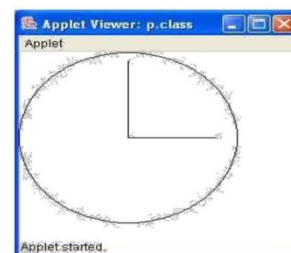
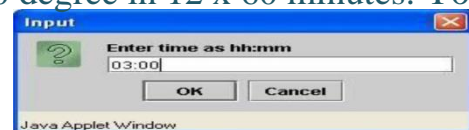
After completing this, the students would be able to:

- ❑ Understand applets in terms of drawing graphical shapes, colors, fonts, drawing complex shapes, reading text input inside applets.

## PROBLEMS

- 1# Write a graphics program that draws a clock face with a time that the user enters in a text field. (The user must enter the time in the format hh: mm, for example, 09:45).

*Hint:* You need to find out the angles of the hour hand and the minute hand. The angle of the hour hand is harder; it travels 360 degree in 12 x 60 minutes. Your output must be in the following format:





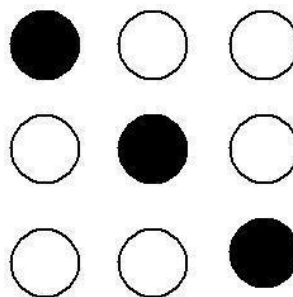
**2#** Write a program that draws the picture of a house.

**Sample Output:**



**3#** Write an applet to display the following figure:

**Sample Output:**



**4#** Draw a "bull's eye" a set of concentric rings in alternating black and white colours: Fill a black circle and then fill a smaller white circle on top, and so on.

