OBJECT ORIENTED PROGRAMMING THROUGH JAVA

LABORATORY MANUAL AY2016_17 E2_SEM1 (BATCH-R15)

INDEX

IIIT RK Valley,
Rajiv Gandhi University of Knowledge Technologies – A.P
516330

OBJECT ORIENTED PROGRAMMING THROUGH JAVA

AY2016-17, SEM1, E2 **LAB: 1 (Basic programs in JAVA)**

CONDITIONAL STATEMENTS AND CONTROL STRUCTURES

1. Write a Java Program to print “Hello World”?
2. Write a Java Program to find sum of two numbers.
3. Write a Java Program to check the given number is prime or not.
4. Write a java program to Check the given year is Leap year or not.
5. Write a java program to find the Sum of digits of given number
6. Write a java program to print the reverse of a given number.
7. Write a java program to find the Factorial of a number.
8. Write a java program to Find maximum of three numbers.
9. Write a java program to find the Power of a number
10. Write a java program to Check the given number is Perfect or not
11. Write a java program to Check the given number is palindrome or not.
12. Write a java program to Check the given number is Armstrong or not.
13. Write a java program for print the characetor ASCII values and read a diferent type of input from user using BufferedReader class
14. Example program on Scanner.
15. Write a java program to print the prime numbers up to the given range n
16. Write a java program to print the Fibonacci series up to the given range n.
17. Write a java program to Print the Perfect numbers in the given range
18. Write a java program to find the GCD of two numbers.
19. Write a java program to find the LCM of two numbers.
20. Write an java program that reads two integers, determines whether the frst is a multiple of the second and print the result.

21. Write a java program to Print the Armstrong numbers up to the given range.

22. Write a java program to print the Perfect squares in the given range.

23. Write a java program to print the prime factors of a given number.

24. Write a java program to print the same digit numbers in the given range.

25. Write a java program to print the Ramanujan numbers from the given range.

IIIT RK Valley,
Rajiv Gandhi University of Knowledge Technologies – A.P
516330

OBJECT ORIENTED PROGRAMMING THROUGH JAVA

AY2016-17, SEM1, E2 **LAB: 2 (Arrays & Strings)**

1. Given a string, return a version where all the "x" have been removed. Except an "x" at the very start or end should not be removed.

E.g.
stringX("xxHxix") → "xHix"
stringX("abxxxcd") → "abcd"
stringX("xabxxxcdx") → "xabcdx"
Note: Write your code in “StringX.Java” file

2. Given an array of strings, return the count of the number of strings with the given length.

E.g.
wordsCount({"a", "bb", "b", "ccc"}, 1) → 2
wordsCount({"a", "bb", "b", "ccc"}, 3) → 1
wordsCount({"a", "bb", "b", "ccc"}, 4) → 0

Note: Write your code in “WordsCount.Java” file

3. Given an array of strings, return a new array containing the first N strings. N will be in the range 1..length.

E.g.
wordsFront({"a", "b", "c", "d"}, 1) → {"a"}
wordsFront({"a", "b", "c", "d"}, 2) → {"a", "b"}
wordsFront({"a", "b", "c", "d"}, 3) → {"a", "b", "c"}

Note: Write your code in “WordsFront.java” file

4. Start with two arrays of strings, A and B, each with its elements in alphabetical order and without duplicates. Return a new array containing the first N elements from the two arrays. The result array should be in

alphabetical order and without duplicates. A and B will both have a length which is N or more. The best "linear" solution makes a single pass over A and B, taking advantage of the fact that they are in alphabetical order, copying elements directly to the new array.

E.g.
mergeTwo({"a", "c", "z"}, {"b", "f", "z"}, 3) → {"a", "b", "c"}
mergeTwo({"a", "c", "z"}, {"c", "f", "z"}, 3) → {"a", "c", "f"}
mergeTwo({"f", "g", "z"}, {"c", "f", "g"}, 3) → {"c", "f", "g"}

Note: Write your code in “MergeTwo.java” file

5. The squirrels in Palo Alto spend most of the day playing. In particular, they play if the temperature is between 60 and 90 (inclusive). Unless it is summer, then the upper limit is 100 instead of 90. Given an int temperature and a boolean isSummer, return true if the
squirrels play and false otherwise.

E.g.
squirrelPlay(70, false) → true
squirrelPlay(95, false) → false
squirrelPlay(95, true) → true

Note: Write your code in “SquirrelPlay.java” file

6. Given three ints, a b c, return true if they are in strict increasing order, such as 2 5 11, or 5 6 7, but not 6 5 7. However, with the exception that if equalOk is true, equality is allowed, such as 5 5 7 or 5 5 5.

E.g.
inOrderEqual(2, 5, 11, false) → true
inOrderEqual(5, 7, 6, false) → false
inOrderEqual(5, 5, 7, true) → true

Note: Write your code in “InOrderEqual.java” file 7. Given three ints, a b c, one of them is small, one is medium and one is large. Return true if the three values are evenly spaced, so the difference between small and medium is the same as the difference between medium and large.

E.g.

evenlySpaced(2, 4, 6) → true
evenlySpaced(4, 6, 2) → true
evenlySpaced(4, 6, 3) → false

Note: Write your code in “EvenlySpaced.java” file

8. Given a string and a non-empty substring sub, compute recursively the largest substring which starts and ends with sub and return its length.

E.g.
strDist("catcowcat", "cat") → 9
strDist("catcowcat", "cow") → 3
strDist("cccatcowcatxx", "cat") → 9

Note: Write your code in “StrDist.java” file

9. Given base and n that are both 1 or more, compute recursively (no loops) the value of base to the n power, so powerN(3, 2) is 9 (3 squared).

E.g.
powerN(3, 1) → 3
powerN(3, 2) → 9
powerN(3, 3) → 27

Note: Write your code in “PowerN.java” file

10. We have bunnies standing in a line, numbered 1, 2, ... The odd bunnies (1, 3, ..) have the normal 2 ears. The even bunnies (2, 4, ..) we'll say have 3 ears, because they each have a raised foot. Recursively return the number of "ears" in the bunny line 1, 2, ... n (without loops or multiplication).

E.g.
bunnyEars2(0) → 0
bunnyEars2(1) → 2
bunnyEars2(2) → 5

Note: Write your code in “BunnyEars.java” file.

11. We'll say that a "pair" in a string is two instances of a char separated by a char. So "AxA" the A's make a pair. Pair's can overlap, so "AxAxA" contains

3 pairs -- 2 for A and 1 for x. Recursively compute the number of pairs in the given string.

E.g.
countPairs("axa") → 1
countPairs("axax") → 2
countPairs("axbx") → 1

Note: Write your code in “CountPairs.java” file.

12. Say that a "mirror" section in an array is a group of contiguous elements such that somewhere in the array, the same group appears in reverse order. For example, the largest mirror section in {1, 2, 3, 8, 9, 3, 2, 1} is length 3 (the {1, 2, 3} part). Return the size of the largest mirror section found in the given array.

E.g.
maxMirror({1, 2, 3, 8, 9, 3, 2, 1}) → 3
maxMirror({7, 1, 2, 9, 7, 2, 1}) → 2

Note: Write your code in “MaxMirror.java” file. 13. Given n>=0, create an array with the pattern {1, 1, 2, 1, 2, 3, ... 1, 2, 3 .. n} (spaces added to show the grouping). Note that the length of the array will be 1 + 2 + 3 ... + n, which is known to sum to exactly n*(n + 1)/2.

E.g.
seriesUp(3) → {1, 1, 2, 1, 2, 3}
seriesUp(4) → {1, 1, 2, 1, 2, 3, 1, 2, 3, 4}
seriesUp(2) → {1, 1, 2}

Note: Write your code in “SeriesUp.java” file.

14. Given a String name, e.g. "Bob", return a greeting of the form "Hello Bob!"

E.g.
helloName("Bob") → "Hello Bob!"
helloName("Alice") → "Hello Alice!"
helloName("X") → "Hello X!"

Note: Write your code in HelloName.java

15. Given a string of even length, return a string made of the middle two chars, so the string "string" yields "ri". The string length will be at least 2.

E.g.
middleTwo("string") → "ri"
middleTwo("code") → "od"
middleTwo("Practice") → "ct”

Note: Write your code in MiddleTwo.java

16. Given a string, if one or both of the first 2 chars is 'x', return the string without those 'x' chars, and otherwise return the string unchanged.

E.g.
withoutX2("xHi") → "Hi"
withoutX2("Hxi") → "Hi"
withoutX2("Hi") → "Hi"

Note: Write your code in WithoutX2.java

17. Return true if the given string contains a "bob" string, but where the middle 'o' char can be any char.

E.g.
bobThere("abcbob") → true
bobThere("b9b") → true
bobThere("bac") → false

Note: Write your code in BobThere.java

18. Given a string and an int N, return a string made of the first N characters of the string, followed by the first N-1 characters of the string, and so on. You may assume that N is between 0 and the length of the string, inclusive (i.e. N>=0 and N <= str.length()).

E.g.
repeatFront("Chocolate", 4) → "ChocChoChC"
repeatFront("Chocolate", 3) → "ChoChC"

repeatFront("Ice Cream", 2) → "IcI"

Note: Write your code in RepeatFront.java

19. Return a version of the given string, where for every star (*) in the string the star and the chars immediately to its left and right are gone. So "ab*cd" yields "ad" and "ab**cd" also yields "ad".

E.g.
starOut("ab*cd") → "ad"
starOut("ab**cd") → "ad"
starOut("sm*eilly") → "silly"

Note: Write your code in StarOut.java

20. We'll say that a "triple" in a string is a char appearing three times in a row. Return the number of triples in the given string. The triples may overlap.

E.g.
countTriple("abcXXXabc") → 1
countTriple("xxxabyyyycd") → 3
countTriple("a") → 0

Note: Write your code in CountTriple.java

21. Given two strings, base and remove, return a version of the base string where all instances of the remove string have been removed (not case sensitive). You may assume that the remove string is length 1 or more. Remove only non-overlapping instances, so with "xxx" removing "xx" leaves "x".

E.g.
withoutString("Hello there", "llo") → "He there"
withoutString("Hello there", "e") → "Hllo thr"
withoutString("Hello there", "x") → "Hello there"

Note: Write your code in WithoutString.java

22. Given 2 arrays of ints, A and B, return true if they have the same first element or they have the same last element. Both arrays will be length 1 or more.

E,g.
commonEnd({1, 2, 3}, {7, 3}) → true
commonEnd({1, 2, 3}, {7, 3, 2}) → false
commonEnd({1, 2, 3}, {1, 3}) → true

Note: Write your code in CommonEnd.java

23. Given an array of ints of even length, return a new array length 2 containing the middle two elements from the original array. The original array will be length 2 or more.

E.g.
makeMiddle({1, 2, 3, 4}) → {2, 3}
makeMiddle({7, 1, 2, 3, 4, 9}) → {2, 3}
makeMiddle({1, 2}) → {1, 2}

Note: Write your code in MakeMiddle.java

24. Given 2 int arrays, a and b, of any length, return a new array with the first element of each array. If either array is length 0, ignore that array.

E.g.
front11({1, 2, 3}, {7, 9, 8}) → {1, 7}
front11({1}, {2}) → {1, 2}
front11({1, 7}, {}) → {1}

Note: Write your code in Front11.java

25. Given an array of ints, return true if the array contains no 1's and no 3's.

E.g.
lucky13({0, 2, 4}) → true
lucky13({1, 2, 3}) → false
lucky13({1, 2, 4}) → false

Note: Write your code in Lucky13.java

26. Given an array of ints, return true if the value 3 appears in the array exactly 3 times, and no 3's are next to each other.

E.g.
haveThree({3, 1, 3, 1, 3}) → true
haveThree({3, 1, 3, 3}) → false
haveThree({3, 4, 3, 3, 4}) → false

Note: Write your code in HaveThree.java

27. Return an array that contains the exact same numbers as the given array, but rearranged so that all the zeros are grouped at the start of the array. The order of the non-zero numbers does not matter. So {1, 0, 0, 1} becomes {0 ,0, 1, 1}. You may modify and return the given array or make a new array.

E.g.
zeroFront({1, 0, 0, 1}) → {0, 0, 1, 1}
zeroFront({0, 1, 1, 0, 1}) → {0, 0, 1, 1, 1}
zeroFront({1, 0}) → {0, 1}

Note: Write your code in ZeroFront.java

28. write a java program that look for patterns like “ram” and “rom” in the string – length should be 3, starting with „r‟ and ending with „m‟. You have to return a string where for all such words, the middle letter is gone, and so “ramXrom” yields “raXrm”.

i. ramRom(“ramXrom”) --␣ “rmXrm”
ii. ramRom(“ramrom”)--␣ “rmrm”
iii. ramRom(“rrromrom”) --␣ “rrrmrm”

Note: write your code in RmPattern.java

29. Given a string that contains a single pair of parenthesis, compute recursively a new string made of only of the parenthesis and their contents, so “xyz(abc)123” yields “(abc)”
Ex:
parentBit(“xyz(abc)123”)--␣ “(abc)”
parentBit(“x(hello)”) --␣ “(hello)”

parentBit(“(xy)1”) --␣ “(xy)”

Note: Write your code in ParentBit.java

30. Returns true if for every „*“ (star) in the string, if there are chars both immediately before and after the star, they are the same

Ex:
sameStar("xy*yzz”)--␣ true
sameStar(“xy*zxx”) --␣false
sameStar(“*xa*az”)--␣ true

Note: write your code in SameStar.java

31. Here I have a string value. You have to split that string into tokens. Store each token into a String array. You have to print all even index values from that array. The String is :”SACHIN TENDULKAR IS MASTER LEGEND”
31. Write a program to substitute the vowels present in a string

i. For example if I enter: I am happy
ii. ii. First it has to search for the vowels and display that the vowels are :I, a,a
iii. Then it has to substitute the vowels by a , e, I, o, u by b, f, j,p,v
iv. The changed string is: j bm hbppy

Input: I am happy
Output: The vowels in given string: I,a,a,
Number of vowels in given string: 3
Changed string: j bm hbppy

32. Write a java program to identify the total number of occurrences of the sub string. Here the String value is: “The problem is not the problem, the problem is your attitude about the problem do you understand!!” Substring is : “the”. You have to identify the total number of occurrences of substring and print the count.

IIIT RK Valley,
Rajiv Gandhi University of Knowledge Technologies – A.P
516330

OBJECT ORIENTED PROGRAMMING THROUGH JAVA

AY2016-17, SEM1, E2 **LAB: 3 (Classes, Objects and Encapsulation)**

1. Create class Number with only one private instance variable as a double primitive type. To include the following methods (include respective constructors) isZero( ), isPositive(), isNegative( ), isOdd( ), isEven( ), isPrime(), isAmstrong() the above methods return boolean primitive type. getFactorial(), getSqrt(), getSqr(), sumDigits(), getReverse() the above methods return double primitive type.

2. Create class SavingsAccount. Usea static variable annualInterestRate to store the annual interest rate for all account holders. Each object of the class contains a private instance variable savingsBalance indicating the amount the saver currently has ondeposit. Provide method calculateMonthlyInterest to calculate the monthly interest by multiplying the savingsBalance by annualInterestRate divided by 12 this interest should be added to savingsBalance. Provide a static method modifyInterestRate that sets the annualInterestRate to a new value. Write a program to test class SavingsAccount. Instantiate two savingsAccount objects, saver1 and saver2, with balances of $2000.00 and $3000.00, respectively. Set annualInterestRate to 4%, then calculate the monthly interest and print the new balances for both savers. Then set the annualInterestRate to 5%, calculate the next month’s interest and print the new balances for both savers.

3. Write a program that stores the details of the Software and Hardware books. The Software book includes the software version and software

name. The Hardware book includes the hardware category and publisher. However, both the books include some common details, such as author name, title, price, and number of pages. Therefore, you need to store and display the book details by implementing the code reusability in the program.

Note: Make all the information as private

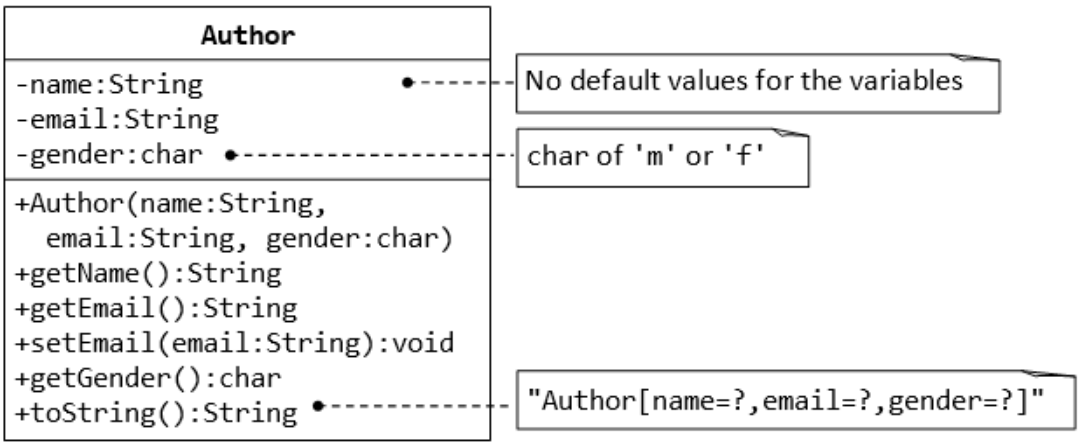
4. Create a class DateTime, the class have a instance members are day(int), month(string), year(int), hour(int), minuets(int) and seconds(int).This class have two parameterized constructor for initialized you data members, one is user for date data member initialization another one for time data members initialization, there is one more constructor is use for store entire information of date and time but the constructor must accept only two object parameters one is date another one is time. Provide a setter and getter method for this data members and one toString method for return the date and time.

IIIT RK Valley,
Rajiv Gandhi University of Knowledge Technologies – A.P
516330

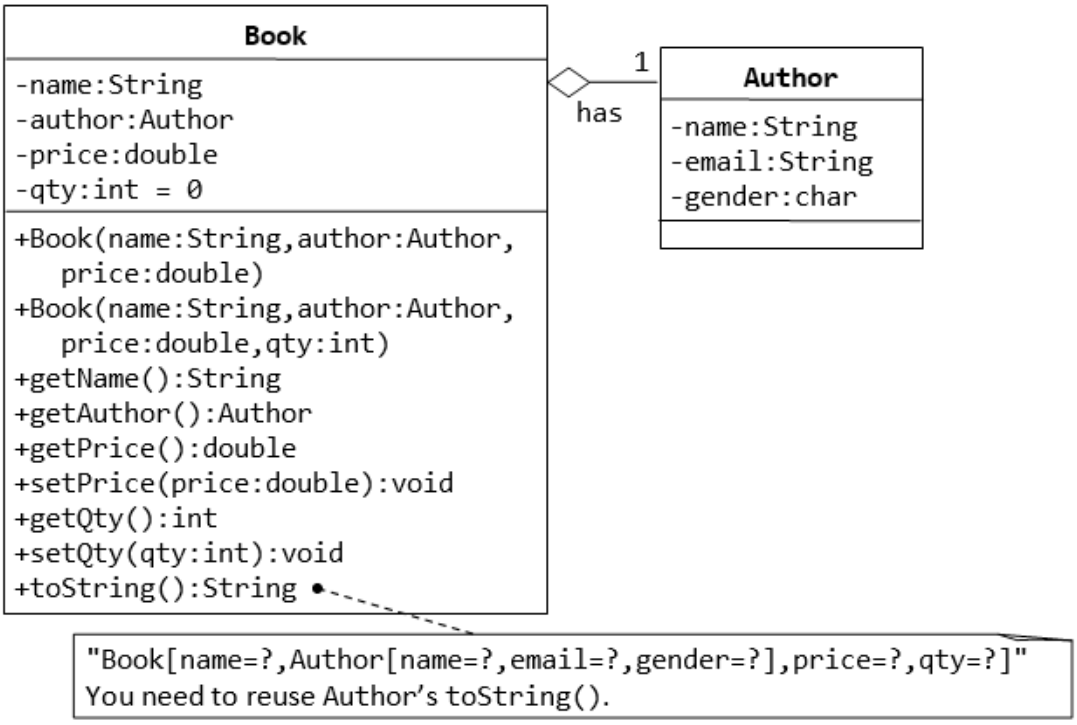
OBJECT ORIENTED PROGRAMMING THROUGH JAVA

AY2016-17, SEM1, E2 **LAB: 4 (Inheritance and Array Objects)**

1. Implement a object oriented programming concept for the given diagram
   A class called Author is designed as shown in the class diagram. It contains: three private instance variables: name (String), email(String), and gender(char of either 'm' or 'f');. One constructor to initialize the name, email and gender with the given values a. Write Public getter/setter: getName(), getEmail(), setMail(), and getGender() b. A toString() method that returns "author-name (gender) at email " eg.rahul (m) at Rahul@gandi.com c. Write the Author class. Also write a test program called TestAuthor to test the constructor and public methods. Try changing the email of an author



2. A class called Book is designed as shown in the class diagram.



It contains: ¬ʌ⊏Four private instance variables: name (String), author (of the class Author you have just created, assume that each book has one and only one author), price (double), and qtyInStock (int); ¬ʌ⊏Two constructors: ¬ʌ⊏public Book (String name, Author author, double price) {...} ¬ʌ⊏public Book (String name, Author author, double price, int qtyInStock) {...} ¬ʌ⊏public methods getName(), getAuthor(), getPrice(), setPrice(), getQtyInStock(),setQtyInStock( ). ¬ʌ⊏toString() that returns "'book-name' by author-name (gender) at email". (Take note that the Author's toString() method returns "author-name (gender) at email".) Write the class Book (which uses the Author class written earlier). Also write a test program called TestBook to test the constructor and public methods in the class Book. Take Note that you have to construct an instance ofAuthor before you can construct an instance of Book. E.g., Author anAuthor = new Author(.....);

Book aBook = new Book("Java for dummy", anAuthor, 19.95, 1000); // Use an anonymous instance of Author Book anotherBook = new Book("more Java for dummy", new Author(.....), 29.95, 888); Take note that both Book and Author classes have a variable called name. However, it can be differentiated via the referencing instance. For a Book instance says aBook, aBook.name refers to the name of the book; whereas for an Author's instance say auAuthor, anAuthor.name refers to the name of the author. There is no need (and not recommended) to call the variables bookName and authorName. TRY: 1. Printing the name and email of the author from a Book instance. (Hint: aBook.getAuthor().getName(), aBook.getAuthor().getEmail()). 2. Introduce new methods called getAuthorName(), getAuthorEmail(), getAuthorGender() in the Book class to return the name, email and gender of the author of the book. For example, public String getAuthorName() { ..... }

3. Create necessary classes for the given scenario ¬ Create a superclass Student, and two subclasses, Undergrad and Grad.
¬ The superclass Student should have the following data members: name, ID, grade, age, and address. ¬ The superclass, Student should have at least one method: boolean isPassed (double grade) The purpose of the isPassed method is to take one parameter, grade (value between 0 and 100) and check whether the grade has passed the requirement for passing a course. In the Student class this method should be empty as an abstract method. ¬ The two subclasses, Grad and Undergrad, will inherit all data members of the Student class and override the method isPassed. For the UnderGrad class, if the grade is above 70.0, then isPassed returns true, otherwise it returns false. For the Grad class, if the grade is above 80.0, then isPassed returns true, otherwise returns false. Create a test class for your three classes. In the test class, create one Grad object and one Undergrad object. For each object, provide a grade and display the results of the isPassed method

4. Write a program to maintain the office database using single inheritance. Superclass is Employee that contain the information as follows- Emp_code, Emp_name, Address, Ph_no, Da 10%, Hra-20%. Employee class can have computeSalary() method to finding the total salary. Create two subclasses of Manager, Typist each class having their own basic pay & da,hra. You have to override computeSalary() method in subclasses 5. Create an abstract class with class name Shape. Create a constructor with arguments and declare variables dim1, dim2 and PI. Declare an abstract method area() inside the class. Create the classes Rectangle, Triangle, Circle, and Ellipse to find the area. Define abstract method area() inside the subclasses and call the

constructor of class Shape using super keyword. In main(), create the objects for all classes and pass values to fields of constructors. Create a reference variable figuref for abstract class. 6. Create class Number with only one private instance variable as a double primitive type. To include the following methods (include respective constructors) isZero( ), isPositive(), isNegative( ), isOdd( ), isEven( ), isPrime(), isAmstrong() the above methods return boolean primitive type. getFactorial(), getSqrt(), getSqr(), sumDigits(), getReverse() the above methods return double primitive type.

5. Write a java program to create a class named Shape that contains A method name numberOfSides(). Provide three classes named Trapezoid, Traingle and Hexagon such that each of the classes extends the class Shape. Each one of the classes contains only the method numberOfSides(). Create a Demo class for checking the number of sides for the Trapezoid, Traingle and Hexagon.

6. Write java code for checking Encapsulation for the given scenario.

Create a course class with courseName, courseId, courseDuration. Make them as private variables. Define appropriate setter and getter methods for accessing outside of class. You have to store atleast 5 course details in an object array of course. You have to display list of courses from object array.

7. Write sample code for creating Person object array to store persons data. Implement Person class with name, age and a toString method. Create necessary classes to store data in object array

8. Write a java program to sort an Integer array.

9. Write a java program to find the frequency of occurrences of each number from the given array

10. Write a java program to sort the String array

11. Write a java program to print the sum of two matrix

12. Write a java program to print the matrix multiplication

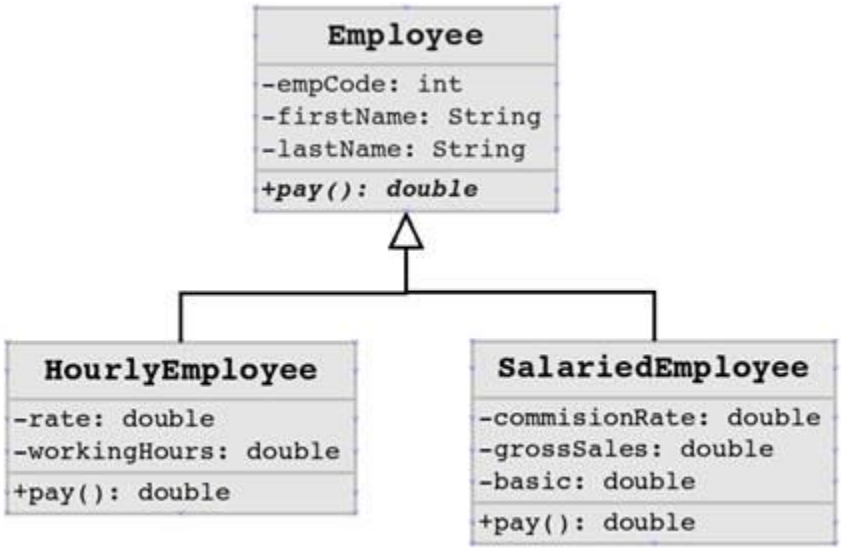
IIIT RK Valley,
Rajiv Gandhi University of Knowledge Technologies – A.P
516330

OBJECT ORIENTED PROGRAMMING THROUGH JAVA

AY2016-17, SEM1, E2 **LAB: 5 (Abstract, Interface and Polymorphism)**

1. Write a Java code for implements Polymorphism in Dynamic method dispatching concept for below diagram

2. Write a Java code for implements Polymorphism in Dynamic method dispatching concept for below diagram



3. Read the Scenario Complete,
   - ➔ Consider we want to start a service like "makemytrip.com" or "expedia.com", where we are responsible for displaying the flights from various flight service company and place an order from customer.
   - ➔ Define BookingObject class you may include you'r own data member, methods and constructor.
   - ➔ Define interface as FlightOpeartions in include the two methods getAllAvailableFlights() and booking(BookingObject bookingObj).
   - ➔ Craete BritishAirways,Emirates and Airasia flight class inherit from BookingObject class and implements from FlightOperation, those class are include the getAllAvailableFlights() to fetch flight details and booking(BookingObject bookingObj) to place order for seat override from FlightOperation interface

   Lets keep our service as simple as,

   a) Displaying flights available from vendors like "airasia", "british airways" and "emirates".

   b) Place and order for seat to respective vendor.

4. Create class SavingsAccount. Use a variable annualInterestRate to store the annual interest rate for all account holders. Each object of the

class contains a private instance variable savingsBalance indicating the amount of the saver currently has on deposit. Provide method calculateMonthlyInterest to calculate the monthly interest by multiplying the savingsBalance by annualInterestRate divided by 12 this interest should be added to savingsBalance. Provide a static method modifyInterestRate that sets the annualInterestRate to a new value. Write a program to test class SavingsAccount. Instantiate two savingsAccount objects, saver1 and saver2, with balances of 2000.00 and 3000.00, respectively. Set annualInterestRate to 4%, then calculate the monthly interest and print the new balances for both savers. Then set the annualInterestRate to 5%, calculate the next month's interest and print the new balances for both savers.

5. Write a program to maintain the office database using single inheritance. Superclass is Employee that contain the information as follows- Emp_code, Emp_name, Address, Ph_no, Da 10%, Hra-20%. Employee class can have computeSalary() method to finding the total salary. Create two subclasses of Manager, TeamLead each class having their own basic pay, da and hra. You have to override computeSalary() method in subclasses. Create a Demo class for checking the computeSalary() method for both Manager and TeamLead objects

1. Write java program for reading input.txt and display the following results
   a. Count total number of words from input.txt
   b. Count total number of characters from input.txt
   c. Count total number of lines from input.txt
   d. Count total number of digits from input.txt
   e. Count total number of spaces from input.txt
   f. Count total number of special characters from input.txt

2. Write a Java program to read data from file and write data into file.
   A simple encryption scheme is to interchange letters of the alphabet on a one-to-one basis. This scheme can be accomplished with a translation table for the lowercase and uppercase letters. Write a program that uses such a scheme to encode text. If you're interested, learn about a more secure encryption system and then program it.
   Note: treat numbers, spaces, and special chars as same.
   Original: a b c d e f g h i j k l m n o p q r s t u v w x y z
   Encryption Table: d e f g h i j k l m n o p q r s t u v w x y z a b c

3. You have to read data from data.txt and encrypt the text. After encryption you have write encrypted text into another file called encrypt.txt. You have to use the above encryption scheme.

4. Implementing the Caesar Cipher

Assignment 1: Word Play

You will write a program to transform words from a file into another for m, such as replacing vowels with an asterix. Specifically, you should d o the following.

● Create a new class called WordPlay.

●Write a method isVowel() that has one parameter named ch. This method returns true if ch is a vowel (one of a, e, i, o, or u or the uppercase versions) and false otherwise.

You should write a tester method to see if this method works correctly. For example, isVowel('F') should return false, and isVowel('a') should return true.

● Write a method replaceVowels()
that has two parameters, a String named phrase and a
character named
ch . This method should return a String that is the string phrase with al
l the vowels (uppercase or lowercase) replaced by
ch . For example, the call replaceVowels(“Hello World”, ‘*’)
returns the string “H*ll* W*rld” . Be sure to call the method isVowel()
that you wrote and also test this method.
● Write a method emphasize with two parameters, a String named
phrase and a character named
ch . This method should return a String that is the string phrase but wit
h the character ch (upper or lowercase) replaced by ○ ‘*’
if it is in an odd number location in the string (first character has index
0, third character has index 2, etc.) ○ ‘+’
if it is in an even number location in the string (second character has i
ndex 1, fourth character has index 3, etc.) For example, the call
emphasize(“dna ctgaaactga”, ‘a’) would return the string
“dn* ctg+*+ctg+” , and the call emphasize(“Mary Bella
Abracadabra”, ‘a’) would return the string
“M+ry Bell+ +br*c*d*br+” . Be sure to test this method.

Assignment 2: Caesar Cipher

You will start with the Caesar Cipher algorithm you learned about in the videos, and you will
make some enhancements to it, so that it works with all letters (both u
ppercase and lowercase) and to make it a little bit harder to decrypt.
Write these methods in a CaesarCipher class you
can use in the next lesson. Specifically, you should do the following: ●
Create a new class called CaesarCipher . ● Write the method encrypt
that has two parameters, a String named input and an int named
key. This method returns a string that has been
encrypted. Assume that all the alphabetic characters are
uppercase letters. For example, the call
encrypt(“FIRST LEGION ATTACK EAST FLANK!”, 23)
should return the string
“CFOPQ IBDFLK XQQXZH BXPQ CIXKH!”

- Write the void method testCaesar() that has no parameter. This method should read a file and encrypt the complete file using the Caesar Cipher algorithm, printing the encrypted message.

You may want to include the lines:

```
FileResource fr = new FileResource();
String message = fr.asString();
String encrypted = encrypt(message, key);
System.out.println("key is " + key + "\n" + encrypted);
```

● Modify the encrypt
method to be able to handle both uppercase and lowercase letters. For example, encrypt("First Legion", 23) should return "Cfopq Ibdflk" and encrypt("First Legion", 17) should return "Wzijk Cvxzfe".
Be sure to test the encrypt method

IIIT RK Valley,
Rajiv Gandhi University of Knowledge Technologies – A.P
516330

OBJECT ORIENTED PROGRAMMING THROUGH JAVA

AY2016-17, SEM1, E2 **LAB: 7 (Exception Handling)**

1. Create an user defined exception called PercentageException with necessary implementation. You have to create a Student class with name, age and percentage with necessary methods. If student percentage is below 50, then you have to raise an exception (PercentageException). If no exception found, then you have to write student name, age, percentage into file
2. Create an user defined exception class called AgeException with necessary implementation. Create a Voter class with name and age. If voter age is below 18 then you have to raise an Exception (AgeException). If age is >=18 then you have to enter voter details into a file. Implement all necessary classes and methods.
3. Create an user defined exception class called NumericException with necessary implementation. You have to create a Person class having name and age. If name is having any number in it, you have to handle that with an user defined exception(NumericException). If no numeric is found in name, then write person name and age into file. Implement necessary classes and methods.
4. Write sample code for the given scenario. You have to create user defined exception class and you have to handle the exception for given scenario. Scenario: create an Exception class with name PrimeException. Create a NumberCheck class and implement a static void checkNum(int number) method. In this method you have to find the given number is prime or not. If the number is prime, then you have to throw the PrimeException, otherwise print the number. Implement the necessary logic for the given scenario.

IIIT RK Valley,
Rajiv Gandhi University of Knowledge Technologies – A.P
516330

OBJECT ORIENTED PROGRAMMING THROUGH JAVA

AY2016-17, SEM1, E2 **LAB: 8 (List Operations)**

1. You have to implement a simple shopping cart problem. In a shopping cart user will have the list of items he added to his cart. For this shopping cart, he can add new item, view items in his list, update an item and remove the item. When the program starts, it should read a shoppingCart.txt file and load all the items into your Shoppingcart. This looks very simple. But you should follow a proper structure to design this program (LinkedList implementation). Framework for shopping cart:

ShoppingCart.java: //Here you have list of CartItem.

You have to implement the following methods

1. Public void add(String productId, String productName, float price, int quantity);

2. public void update(String prodId,int quantity);

3. public void display();

4. public void remove(String prodId);

CartItem.java:

This class contains 4 members.

1. String productId; 2. String name; 3. float price; 4. int quantity = 0;

Define a constructor with necessary parameters. Implement the toString method to get the cartItem information.

ShoppingCartMenu.java

Public class ShoppingCartMenu {

Public static void main(String args[]){ // Here you have to read cart items from items.txt and send items to shopping cart // Implement the menu interface to add new item, update the quantity of product, display all Items, remove item and exit. } } Note: Strictly follow the above framework.

2. You have to implement a simple application for Mobile shop. Mobile shop has list of Mobiles. For this Mobile shop, you can add new mobile, view all mobiles in shop, update mobile and remove the mobile. When the program starts, it should read mobile details from mobiles.txt file and load all the items to Mobile shop. This looks very simple. But you should follow a proper structure to design this program (LinkedList implementation). Framework for shopping cart:

MobileShop.java: //Here you have list of Mobiles.

You have to implement the following methods

1. Public void add(String company_Name, String model_Name, float price);

2. public void update(String model_Name);

3. public void display();

4. public void remove(String Model_Name);

Mobile.java:

This class contains 4 members.

1. String company_Name; 2. String model_Name; 3. float price;

Define a constructor with necessary parameters. Implement the toString method to get the cartItem information.

MobileShopInventory.java

Public class MobileShopInventory { Public static void main(String args[]){ // Here you have to read mobiles data from mobiles.txt and send items to mobile shop // Implement the menu interface to add new mobile, update mobile model, display all mobiles, remove mobile and exit. } }

Note: Strictly follow the above framework.

3. You have to implement a simple application for Amazon store. Store has to maintain list of Laptops. You can add new Laptop, view all Laptops in store, update Laptop details and remove the Laptop from store. When the program starts, it should read all Laptop details from stock.txt file and load all the items to Amazon store. This looks very simple. But you should follow a proper structure to design this program (LinkedList implementation). Framework for AmazonStore_App:

AmazonStore.java: //Here you have list of Laptops.

You have to implement the following methods

1. Public void add(String model_Name, String serial_No, float price);

2. public void update(String serial_No);

3. public void display();

4. public void remove(String serial_No);

5. public void sort() // sort based on price

Laptop.java:

This class contains 4 members.

1. String model_Name; 2. String serial_No; 3. float price; 4. Laptop

Define a constructor with necessary parameters. Implement the toString method to get the Laptop information.

AmazonStoreApp.java

Public class AmazonStoreApp { Public static void main(String args[]){ // Here you have to read all Laptop data from stock.txt and send items to Amazon store // Implement the menu interface to add new Laptop, update Laptop model, display all Laptops, sort laptops based on price, remove Laptop and exit. } } Note: Strictly follow the above framework.

1. Tollywood Film industry wants to rearrange the movie details in a file. Here you have the movie details in a file. These details are not in order. You have to rearrange these details in decreasing order based on gross collections. You have to read movie details from file and keep movie details in a data structure (Linked List). User can search for a movie based on movie name; user can delete a movie based on movie name. User can sort the movies based on gross collections. At the end user must store sorted movie details in a file. Here I am providing Movie class. Based on the movie class, you have to create MovieList class and implement all methods. Create a Demo class for getting movie details from file.
   Class MovieInfo {
   String movie_Name;
   String hero_Name;
   String director_Name
   Int release_Date; float gross_Collections; MovieInfo next; //TO-DO create a constructor to values for instance variables //TO-DO Create a toString method for returning movieInfo }

5. BSNL have the all subscribers' data with them. BSNL provides data file to you. You have to sort the file data based on subscriber id. You have to read subscriber details from file and keep details in a data structure (Lined List). User can search for a subscriber based on subscriber id. User can delete subscriber based on subscriber id. User have to sort the subscribers based on subscriber id. At the end user have to store subscribers details in a file. Here I am providing subscriber class. Based on subscriber class you have to create subscriberList class and implement all methods. Create a Demo class for getting subscriber details from file Class SubscriberInfo { String subscriber_Id String subscriber_Name; Int balance; SubscriberInfo next; //TO-DO Create a constructor to assign values for instance variables //TO-DO Create a toString method for returning SubscriberInfo }

6. BCCI wants to select team players based on their average score. BCCI provides data file to you. You have to sort the player based on their average score. You have to read player details from file and keep details in a data structure(Linked List). User can search for a player based on name. user can delete player based on player name. user have to sort the players based on average score. At the end user have to store players details in a file. Here I am providing player class. Based on player class you have to create playerList class and implement all methods. Create a demo class for getting player details from file. Class Player { String playerName; Int totalRuns; Float average; Player next; //TO-DO Create a constructor to assign values for instance variables //TO-DO Create a toString method for returning Player info }

IIIT RK Valley,
Rajiv Gandhi University of Knowledge Technologies – A.P
516330

OBJECT ORIENTED PROGRAMMING THROUGH JAVA

AY2016-17, SEM1, E2 **LAB: 9 (Multi Threading)**

Objective: Our aim is to make you understand the concept of Multi-Threading

Scenario – 1

Problem Definition:

- Developing an University Registration System where students register for admission - For registration process student should contact the university registrar. - The university had appointed a Registrar and a Validator. They are simultaneously accepting the student details for registration. - Registrar will accept the student details and intimate to Validator. Registrar will wait for Validator's intimation. - Validator get the student details and validates and throws AgeException if student age is <35 and >23. If age is valid, students details are write in to a file. After that validator intimate to Registrar for next student details. - Create a file named StudentDetails.txt to store the details of student - You have to create a Student class with name, age and implement necessary methods like push, pop, toString. - Create Registrar, Validator thread classes for simultaneously accessing the student details.

Scenario – 2

Problem Definition:

- Developing a Department Out Pass System where students will get out pass by ASWO and verified by HOD. - Department ASWO will verify the student outpass details of roll_No and max_Leave_Days and reason for leave. - If max_Leave_Days are above 5 days, ASWO will reject student outpass.

- After processing the student outpass by ASWO, he will wait for HOD instruction. if outpass will get by HOD, he will write student outpass details like student roll_no, max_Leave_Days and reason for leave into file. He will notify to ASWO for next outpass. - After entering details into file, then only ASWO will process another student details to HOD. - You have to prepare a studentOutpass class with necessary variables and implement methods like push, pop, toString. - Create ASWO, HOD thread classes for simultaneously accessing the student leave applications. - Create a Student_Leave class where student apply for leave.

Scenario – 3

Problem Definition:

- Developing a Hotel InBook Log System where customer meets the Receptionist and submit details to Receptionist, after that details collected by Book keeper and write into a text file. - Receptionist will collect Customer data and Intimate to Book keeper. Book keeper will get the details and write into file. After write into file, book keeper will intimate to Receptionist. Receptionist will collect 5 customer details and Book keeper will write each customer details in file. - Customer has name, age and mail_id. You have to implement push, pop and toString methods in customer class. - Create Receptionist, Book keeper thread classes and implement accordingly. These objects are simultaneously accessing the customer object. - Create a Demo class for running the program.

4. Create two thread classes. One thread class will generate Even number and another thread class will write that number into file. Both thread should be communicate through predefined methods like notify(), and wait(). Producer thread has to insert number into a buffer and notify the consumer thread. Consumer thread will get the value from buffer, it will send value to

file and notify to producer for next number. Implement the necessary classes and methods to achieve the multi-threading.

5. Create two thread classes. One thread class will generate prime number and another thread class will write that number into file. Both thread should be communicate through predefined methods like notify(), and wait(). Producer thread has to insert number into a buffer and notify the consumer thread. Consumer thread will get the value from buffer, it will send value to file and notify to producer for next number

6. Create two thread classes. One thread class will generate odd number and another thread class will write that number into file. Both thread should be communicate through predefined methods like notify(), and wait(). Producer thread has to insert number into a buffer and notify the consumer thread. Consumer thread will get the value from buffer, it will send value to file and notify to producer for next number. Implement the necessary classes and methods to achieve the multi-threading

IIIT RK Valley,
Rajiv Gandhi University of Knowledge Technologies – A.P
516330

OBJECT ORIENTED PROGRAMMING THROUGH JAVA

AY2016-17, SEM1, E2 **LAB: 10 (AWT Programming)**

1. Develop a GUI that receives an integer in one text field, and computes its factorial Value and returns it in another text field, when the button named “Compute” is clicked?
2. 2. Develop a GUI that receives Student info like Student Name, Age, Mail and writes data into file, when the button name “Register” is clicked?