Object Oriented Programming Using Java (18CSI302)

QUESTION BANK

Unit 1: Introduction to Object Oriented Programming

1. State whether the following statements are TRUE or FALSE.

(a) The main emphasis of procedure oriented programming is on algorithms rather than data.

(b) One problem with OOP is that once a class is created, it can never be changed.

(c) Object oriented approach cannot be used to create databases.

(d) Object oriented systems can scale up better from small to large.

(e) Wrapping up of data of different types into a single unit is known as encapsulation.

(f) In conventional, procedure-oriented programming, all data are shared by all functions.

(g) Polymorphism is extensively used in implementing inheritance.

2. What are command line arguments? Explain with suitable example.
3. Describe the characteristics of object oriented programming concepts?
4. Write the Differences between Java and C++.
5. Describe the features (buzzwords) of Java Programming language?
6. The factorial of an integer m is the product of consecutive integers from 1 to m. That is, m!=m*(m-1)*….*1. Write a program that computes and prints a table of factorials for any given m.
7. Write a program to compute the sum of the digits of a given integer number.
8. The numbers in the sequence 0 1 1 2 3 5 8 13 21 …. are called as Fibonacci numbers. Write a program using a do-while loop to calculate and print first m Fibonacci numbers.
9. Explain various data types available in java?
10. Distinguish between procedural language and OOPs?
11. Explain how polymorphism is supported in Java with a suitable example.
12. Write a Java program that mimics a calculator. Input two real numbers and state the type of arithmetic operation (addition, subtraction, multiplication and division) between them through another keyboard input.
13. List and explain the characteristic features of java language.
14. Explain the process of building and running java application program.
15. Write a java program to multiply two 3 X 3 matrices.
16. Elaborate on the various Object Oriented concepts, with necessary illustrations.

17. With relevant examples explain abstraction and encapsulation. Write a Java program that uses abstraction and encapsulation. Give self-explanatory comments in your program.
18. Write a java program to reverse a number.
19. Explain with the help of a program how object oriented programming over comes the shortcomings of procedure oriented programming.
20. With example explain the working of >>and >>>.
21. Define type conversion, typecasting with an example?
22. Write a java program to find maximum and minimum values in an array.
23. Explain break and continue statements with an example.
24. Write a notes on operators in java with an example.

Unit 2: Defining your own classes, Objects and Methods

1. Explain the general syntax of defining a class and objects with a suitable example.
2. What is a method? How is a method defined? Give an example.
3. What is constructor in Java? Why constructor does not have return type in Java? Explain it with proper example.
4. Explain Pass by Value and Pass by reference with a suitable example.
5. Explain different access specifiers is Java, with examples.
6. Differentiate between a constructor and a method with an example.
7. State the purpose of finalize() method in Java? With an example how finalize() method can be used in a Java program.
8. With an example, describe in detail about how polymorphism plays a useful role in JAVA.
9. Write a Java program using classes to perform linear search on an array of N numbers. Give self-explanatory comments in your program.
10. Define polymorphism. Show how compile time polymorphism is achieved in a Java program with a suitable example.
11. Why do we need static members and how to access them? Explain it with an example.
12. Explain the following static, finalize and this keywords.
13. How garbage collection works in java.
14. Write Uses of ‘this’ keyword.
15. Explain recursion with an example.
16. Explain default constructor with suitable example.
17. Illustrate parameterized constructor with example.
18. State the use of constructor and finalize() method in Java using a programming example. Show how garbage collection is achieved here.
19. With a programming example, explain constructor overloading.

20. Explain method overloading with a suitable example.
21. Write a java program to check prime number between 1 to 100.
22. Explain static nested class and inner class with examples.
23. Write a program to perform the following functions using classes and objects:
    a) Input the marks of 5 students in 5 subjects.
    b) Calculate the total and average.
    c) Print the formatted result on the screen.
24. Explain the following terms with an example:
    (a) Member nested class (Non-Static)
    (b) static nested class
    (c) Anonymous class

Inheritance, Polymorphism, Abstraction

1. What is inheritance? How will you call parameterized constructor and overrided method from parent class in sub class?

2. Explain with an example how single level inheritance is achieved in Java.

3. Explain with an example how multilevel inheritance is achieved in Java.

4. Can Java directly support multiple inheritance? Illustrate your answer with an example Java program.

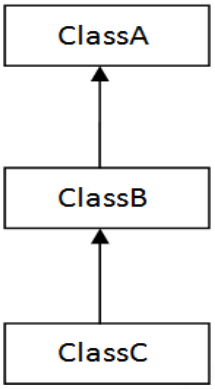
5. Explain the concept of inheritance and its types.

6. Explain the concept of method overriding with examples.

7. Describe the role of the final keyword and give a list of guidelines for when it should and when it should not be used.

8. Describe hierarchical inheritance with an example.

9. Write a program to demonstrate the multilevel inheritance for the classes having relations as shown in figure.



10. With the help of a program explain ”The Diamond Problem” in java.

11. Differentiate between abstract class and interface with an example.

12. With the help of java program, illustrate the usage of super key word to access super class members.

13. Write a short note on Object Class.

14. Illustrate Dynamic method dispatch with an example program.

15. Describe the rules for creating an abstract class in java.