CBCS SCHEME

USN | | | | | | | | | | |

BCS306A

Third Semester B.E./B.Tech. Degree Examination, June/July 2024
Object Oriented Programming with Java

Time: 3 hrs. Max. Marks: 100

Note: 1. Answer any FIVE full questions, choosing ONE full question from each module.
2. M : Marks , L: Bloom's level , C: Course outcomes.

		Module – 1	M	L	C
Q.1	a.	Differentiate two paradigms of programming.	5	L2	CO1
	b.	Explain the various bitwise and short circuit operators in Java.	8	L2	CO1
	c.	Write a Java program with a method to check whether a given number is prime or not.	7	L3	CO1
		OR			
Q.2	a.	Explain various scopes of variables in Java.	5	L2	CO1
	b.	Explain the arithmetic compound assignment and Ternary operators in Java.	8	L2	CO1
	c.	Write a Java program to perform linear search on an array elements accepted from keyboard and key element also accepted from key board.	7	L3	CO1
		Module – 2			
Q.3	a.	Explain method overloading in Java with examples.	8	L2	CO2
	b.	Design a stack class to hold maximum of N numbers with a constructor, push, POP and Display methods. Develop Java main method to illustrate stack operations.	12	L3	CO2
		OR			
Q.4	a.	Explain the role of "this" keyword and "static" keyword in Java.	8	L2	CO2
	b.	Design a class called "Employee" with fields ID, Name and Salary. Write a suitable constructors a method to raise salary and a static method to display. The number of Employee objects. Write suitable Main method for illustration.	12	L3	CO2
		Module – 3			
Q.5	a.	Explain the role of "Super" with example Java program.	6	L2	CO3
	b.	For any class and any method as an example, explain method overriding.	5	L2	CO3
	c.	Develop a Java program to create class called "Shape". Create 3 sub classes : circle, triangle and square. Each class has 2 member function area ( ) and draw ( ). Demonstrate polymorphism with a suitable main program.	9	L3	CO3
		OR			
Q.6	a.	Explain the order of constructor execution in a multilevel class hierarchy.	6	L2	CO3
	b.	Define dynamic method dispatch and write a code snippet in Java to demonstrate.	5	L1	CO3

	c.	Develop Java program to create interface Resizable with methods resize width (int width) and resize height (int height) that allow object to be resized. Create a class Rectangle that implements This Interface.	9	L3	CO3
		Module – 4			
Q.7	a.	Explain four categories of visibility for class members based on packages.	6	L2	CO4
	b.	Give the general form of an exception handling block and write a Java program to illustrate multiple catch classes.	7	L2	CO4
	c.	Write a custom exception in Java called "less marks" and raise This exception when marks entered by valuator in the range [30 – 34]	7	L3	CO4
		OR			
Q.8	a.	With code snippets, explain mechanism to create and import a package in Java.	6	L2	CO4
	b.	Write a Java program to create chained exceptions with top-level exception is Null Pointer Exception and its cause Arithmetic Exception.	7	L3	CO4
	c.	Develop a Java program to create custom exception for Negative odd numbers.	7	L3	CO4
		Module – 5			
Q.9	a.	Explain various methods of thread class in Java.	6	L2	CO5
	b.	Write a Java program to create 4 threads and each thread when run, will sleep for 500 milliseconds and print its name before "Before Quitting".	8	L3	CO5
	c.	Explain the use of Type wrappers in Java with example.	6	L2	CO5
		OR			
Q.10	a.	Explain is Alive ( ) and join ( ) methods of Thread with example code snippet.	6	L2	CO5
	b.	Write a Java program to create 4 Rread and each Thread generates random number and prints it and sleeps for 800 msec and quits.	8	L3	CO5
	c.	Explain the concept of autoboxing /unboxing in expressions and methods.	6	L2	CO5

* * * * *

Visvesvaraya Technological University

Belagavi, Karnataka - 590 018.

BCS306A

Scheme & Solutions

Signature of Scrutinizer

Subject Title : Object Oriented Programming with JAVA **Subject Code :** BCS306A

Question Number	Solution	Marks Allocated
1a	Two paradigms of programming: Procedure oriented Vs object oriented. Any 5 differences – operations Vs objects, top down Vs bottom up, flexibility, reusability, scalability [1 M each]	05
b.	Bitwise operators – &, |, ~, ^, <<, >>, >>> – 1M Short circuit operators → &&, || → 2M	08

c.	class Prime
	{
	boolean checkprime(int num)
	{
	for(int i=2; i<num/2; i++)
	if(num%i==0) return true;
	return false;
	}
	method – 02M, main – 02M
		07

2a.	Various scopes → class scope (2M) and block scope (2M) – 05M method scope – 2M	05
b.	Arithmetic operators → +, -, *, /, % – 4M Compound assignment operators → +=, -=, *=, /= – 2M Ternary operator → cond ? true : false – 2M	8

Subject Title : Object Oriented Programming with JAVA Subject Code : BCS306A

Question Number	Solution	Marks Allocated

2a.	`void readArray(int arr[], int N)`
	`{ Scanner kb = new Scanner(System.in);`
	`for(i=0; i<N; i++)`
	`{ arr[i] = kb.nextInt();`
	`}`
	`}` — 2M
	`boolean reverseLinearSearch(int arr[], int key, int N)`
	`{ for(int i=N-1; i>=0; i--)`
	`{ if(arr[i]==key) return true;`
	`}`
	`return false;`
	`}` — 3M
	main pgm – 2M
		07

3a	Method overloading is a concept where methods have same name but different parameters (type/number). → 03M Eg: `void test(int a){...}` `void test(int a, int b){...}` `void test(double c){...}` — 01M	03.

b.	`class Stack` → 01
	`{ int top; int N; int[] data;` — 01
	`Stack(int N)`
	`{ top=-1;`
	`this.N=N;`
	`data=new int[N];`
	`}` — 02
	`void push(int ele)`
	`{ if(top==N-1) S.o.p("overflow");`
	`data[++top]=ele;`
	`}`
	`void pop(){ if(top==-1) S.o.p("underflow");`
	`S.o.p(data[top--]);}` — 02
	`void display()`
	`{ for(int i=top; i>=0; i--)`
	`S.o.p(data[i]);`
	`}` — 02
	main → 02
		12

Subject Title Object Oriented Programming with JAVA **Subject Code :** BCS306A

Question Number	Solution	Marks Allocated
4a)	This used to resolve instance variable hiding } 04 It refers the current object called upon static → refers class variables and used inside static methods. One copy for entire class } 04	08
b.	Class Employee → 01 { int ID; String name; int salary; static int ecnt=0; Employee (int ID, String name, int salary) { This.ID=ID; ecnt++; This.name=name; This.salary=salary; } static void disp() { S.O.P(ecnt); } 2 void raiseSal (double per) { salary = salary + (int) per * salary; } 2 main — 02M	12
5a.	Super is used to call superclass constructor 2 Class A { . . . } Class B extends A { B() { super(); → calls A } } 04 Other methods of superclass can also be called — 02M	06
b.	Class Shape (02) { void disp () { S.O.P ("I am shape"); } } ← Override → Class Circle extends Shape (03M) { void disp () { S.O.P ("I am circle"); } }	05

Subject Title : Object oriented Programming with JAVA Subject Code : BCS306A

Question Number	Solution	Marks Allocated
5c.	Class Shape { . . } — 07 Class Circle extends Shape { double radius; double area() { return Math.PI * radius * radius; } } Class Triangle extends Shape { double b, h; double area() { return 0.5 * b * h; } } Class Square extends Shape { double side; double area() { return side * side; } } } 06 02 M for main	09
6a.	In case of multilevel hierarchy innermost constructor called first and outermost constructor called last. → 02 Eg program with print – 04 M	06
b.	Dynamic method dispatch is a scenario in which subclass reference can be assigned to superclass and based on specific subclass reference at that instance, appropriate method is invoked – 02 M Eg 03 M	05
c.	interface Resizable { void resizeWidth(int width); void resizeHeight(int height); } 04 Class Rect implements Resizable { } → Code – 05 M	09
7a.	4 types of visibility → same class same package (private) ~~Different~~ Inherited class same package (protected) ~~Same~~ diff class same package (package) any package (public) → 04 + Eg 02 M	06
b.	try { . . } catch(..) { .. } finally { .. } – 03 M – 04 M (Eg)	07

Subject Title : **Subject Code :**

Question Number	Solution	Marks Allocated
7c.	class LenMasks extends Exception → 01 { ... } 04M { if marks >=30 && marks<=34 Throw LenMasks(..)	07
8a.	Package Statement to create package and import Statement to import a Package — 02M Eg: 04M	06
b.	NullPointerException → ArithmeticExceptions Checking → 04 + 03 = 07M	07
c.	class Negativeodd extends Exceptions — 03 { ... } if (num%2!=0 && num<0) } 04 Throw Negativeodd(..)	07
9a.	Thread methods → isAlive(), join(), wait(), suspend(), sleep() etc. }	06M
9b.	class T1 implement Runnable → 04M { public void run() { this.sleep(500); S.o.p("name" + i); } } main method - 04M	08
9c.	Typewrappers - Type conversion Integer, Float etc - (6 wrappers)	06
10a.	isAlive() → 03M join() → 3M → wait for thread to exit Check Thread alive or not	06
b.	class T1 implements Runnable → 05 Main - 03M { public void run() { Random r=new Random(); int n=r.nextInt(); S.o.p(n); } }	08
c.	Autoboxing/unboxing in methods - 03M and expressions (auto type promotions) - 03M	06

End