**Section-1 Question Bank**

**Fill in the blanks/ True or False/ MCQ**

1. Which of the following are valid identifiers?

i) abc ii) Student iii) class iv) public v)1num vi) result

2. What will be the output of the below code?

public static void main(String args[]) {

int age;

System.out.println(age);

}

i)garbage value ii) 0 iii) compilation error : variable needs to be initialized before use iv)2147483647

3. Which of the following are valid variable declarations?

float = 10.5f

boolean isOn = “true”

char gender = ‘F’

int value = 20

4. Which of these values can a boolean variable contain?

true

false

0

1

5.

5. In Java, an \_\_\_\_\_\_\_\_\_ is the name given to a variable, method or class to uniquely identify it.

6. In Java, the identifier name: (Choose the correct option)

* **Can/can’t** contain alphanumeric characters([a-z], [A-Z], [0-9]), dollar sign ($), underscore (\_)
* **Should/should not** start with a digit ([0-9])
* **Should/should not** have spaces
* **Should/should not** be a Java keyword
* It **is/isn’t** case-sensitive
* It **has/has not** length restrictions

7. What will be the output of the below code?

public class Demo {

public static void main(String args[]) {

int a = 10;

int b = 2;

System.out.println((a < b) ? a++ : --b);

}

}

1. 10 b) 2 c) 1 d) 11

8. What will be the output of the below code?

public static void main(String args[]) {

int i = 10;

if(i) {

System.out.println("Hello");

}

else {

System.out.println("Bye");

}

}

Hello

Bye

compilation error: cannot convert from int to boolean

runtime error

**9. What is the order of precedence (highest to lowest) of the below operators?**

&&, ++ (post increment), =, !

++ , ! , && , =

++ , && , ! , =

! , ++ , && , =

++ , = , && , !

**10. A \_\_\_\_\_\_\_\_\_ in Java is a special method that is used to initialize class variables at the time of object creation.**

**11. Constructors have the same name as that of the class and does not have a return type.(True/False)**

**12. If you don’t define a constructor in a class, then Java creates a default parameterless constructor and initializes the default values to the class variables based on the data type. (True/False)**

**13. You can also create parameterless constructor in a class. In this case, Java creates/ does not create a separate default constructor.**

**14. What will be the output of the below code?**

class Demo {

public int value = 20;

Demo() {

value = 40;

}

}

public class Tester {

public static void main(String args[]) {

Demo demo = new Demo();

System.out.println(demo.value);

}

}

1. 20 b) 40 c) 0 d) compile Time Error

**15. What will be the output of the below code?**

class Demo {

public int var1 = 20;

public int var2 = 40;

Demo(int v1, int v2) {

var1 = v1;

var2 = v2;

}

}

public class Tester {

public static void main(String args[]) {

Demo demo = new Demo();

System.out.println(demo.var1);

System.out.println(demo.var2);

}

}

1. 20 40
2. 0 0
3. 40 20
4. compilation error: class Tester has no default constructor

**16. Consider the below Circle class.**

public class Circle {

public double radius;

Circle(double r) {

radius = r;

}

}

Which of the following options will create an object for the class

Circle c = new Circle(10.5)

Circle = new Circle(10.5)

Circle c = new Circle()

Circle c = new Circle(5.4, 10)

**17. Which of the following is/are true about constructors in Java?**

The name of the constructor should be same as the class name.

If you don't define a constructor for a class, a default parameterless constructor is automatically provided by the compiler.

The default constructor initializes all the instance variables to default values of the data types.

A constructor cannot have an access modifier.

**18. What will be the output of the below code?**

public class Tester {

public int num1;

public int num2;

Tester(int num1, int num2) {

num1 = num1;

num2 = num2;

}

public static void main(String args[]) {

Tester tester = new Tester(20, 40);

System.out.println(tester.num1);

System.out.println(tester.num2);

}

}

1. 20,40 b) 0,0 c) Compile time error d)None of the mentioned

19. An array is a collection of non similar data in contiguous memory locations referred by the same name (True/False).

20. Array can be used to store data of primitive as well as reference types (True/False)

21. Array holds a fixed number of values, determined at the time of array\_\_\_\_\_\_\_\_\_\_\_

a) compilation b) execution c)declaration d) processing

22. Array index always starts from \_\_\_\_\_\_

23. Once initialized, the size of an array **can/cannot** be changed

24. Which of the following are valid array declarations?

1) int myArray1[5];

2) int myArray2[];

3) int myArray3[]=new int[5];

4) int myArray4[5]=new int[5];

5) int []myArray5=new int[5];

6) int myArray6[]=new int[];

7) int myArray7[]=null;

25. What will be the output of the below code?

public class Tester {

public static void main(String args[]) {

int arr[] = new int[] { 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 };

int n = 6;

n = arr[arr[n] / 2];

System.out.println(arr[n] / 2);

}

}

1. 3 b) 6 c) 0 d) 1

26. public class Tester {

public static void main(String s[]) {

int a[] = { 12, 15, 16, 17, 19, 23 };

for (int i = a.length - 1; i > 0; i--) {

if (i % 3 != 0) {

--i;

}

System.out.println(a[i]);

}

}

}

1. 23 19 16 15
2. 23 19 15
3. 19 17 15
4. 19 17 15 12

27. Predict the output of the following code:

class Customer {

public Customer() {

System.out.println("Customer");

}

}

class RegularCustomer extends Customer {

public RegularCustomer() {

System.out.println("Regular Customer");

}

}

class PremiumCustomer extends RegularCustomer {

public PremiumCustomer() {

System.out.println("Premium Customer");

}

}

public class Tester {

public static void main(String[] args) {

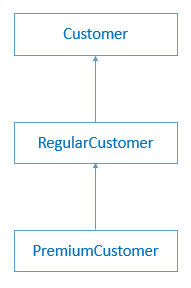
PremiumCustomer premiumCusotmer = new PremiumCustomer();

}

}

1. Customer Regular Customer Premium Customer
2. Premium Customer
3. Regular Customer
4. Customer

28. Following image shows which type of inheritance ?



1. Multiple b) multilevel c) single d) None of the mentioned

29. When more than one class extends the same base class, then that type of inheritance is said to be \_\_\_\_\_\_\_\_\_\_\_Inheritance.

30. \_\_\_\_\_\_\_\_\_\_\_ Inheritance is NOT supported in many object oriented programming languages including Java.

31. Polymorphism that gets resolved during compile time is known as \_\_\_\_\_\_\_ polymorphism or \_\_\_\_\_\_\_\_\_\_\_polymorphism.

32. \_\_\_\_\_\_\_\_\_ polymorphism is achieved using overloading of the methods in the same class, called as Method overloading.

33. What will be the output of the following code?

public class Tester {

public static void main(String[] args) {

Tester tester = new Tester();

tester.evaluate(4, 2.5f);

}

public void evaluate(int a, int b) {

System.out.print("First evaluate method invoked");

System.out.println(a + b);

}

public void evaluate(float a, float b) {

System.out.print("Second evaluate method invoked");

System.out.println(a + b);

}

}

1. First evaluate method invoked 6
2. Second evaluate method invoked 6.5
3. Compilation error as two methods with same name cannot exist in the same class
4. First evaluate method invoked 6.5

**34. Find the output of the code.**

public class Employee {

private String name;

private int empId;

private String city;

Employee() {

this.city = "New York";

}

Employee(String name, int empId) {

this.name = name;

this.empId = empId;

}

public static void main(String[] args) {

Employee employee1 = new Employee("John", 101);

Employee employee2 = new Employee();

System.out.println(employee1.name + " " + employee1.empId + " "

+ employee1.city);

System.out.println(employee2.name + " " + employee2.empId + " "

+ employee2.city);

}

}

1. John 101 New York John 101 New York
2. John 101 null null 0 New York
3. John 101 New York null 0 New York
4. John 101 null John 101 New York

35. Find the output of the following program

class Person {

String name;

Person() {

name = "John";

}

public void getDetails() {

System.out.println(name);

}

}

class Employee extends Person {

int age;

Employee() {

age = 34;

}

public void getDetails() {

System.out.println(name + " " + age);

}

}

class Customer extends Employee {

int salary;

Customer(int salary) {

this.salary = salary;

name = "Maddy";

}

public void getDetails() {

System.out.println(name + " " + age + " " + salary);

}

}

public class Account {

public static void main(String[] args) {

Person c = new Customer(20000);

c.getDetails();

}

}

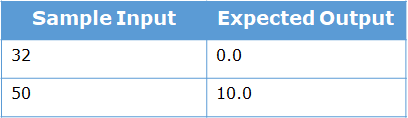
1. John
2. Maddy 34 20000
3. John 34 20000
4. John 34

**Solve the following**

**1. Implement a program to convert temperature from Fahrenheit to Celsius degree by using the formula given below and display the converted value.**

C = ((F-32)/9)\*5 where, C represents temperature in Celsius and F represents temperature in Fahrenheit.

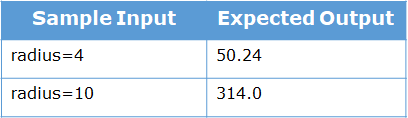
Sample Input and Output



**2. Implement a program to find the area of a circle by using the formula given below and display the calculated area.**

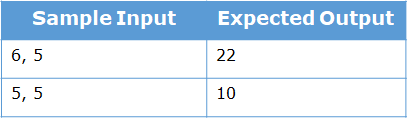
Area = pi\*radius\*radius. The value of pi is 3.14.

Sample Input and Output



**3. Implement a program to display the sum of two given numbers if the numbers are same. If the numbers are not same, display the double of the sum.**





**4. Create a new Java project with "AddressDetails.java" file and implement a Java code to display your address.**

**Sample Output**

Door No: D089

Street: St. Louis Street

City: Springfield

ZIP Code: 62729

5. **Problem Statement**

Modify the Order class created before and add two constructors in the class.



**Method Description**

**Order()**

* Set the value of status to 'Ordered'.

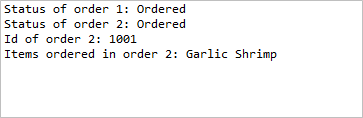
**Order(int orderId, String orderedFoods)**

* Initialize the instance variables appropriately with the values passed to the constructor.
* Set the value of status to 'Ordered'.

Create an object of the Order class by using the parameterless constructor and display the value of the status instance variable in the main() method of the Tester class.

Create one more object of the Order class by using the parameterized constructor and display the value of orderId, orderFoods and status instance variables in the main() method of the Tester class.

**Sample Output**



6. **Problem Statement**

Modify the Restaurant class created before and add the below mentioned constructor.



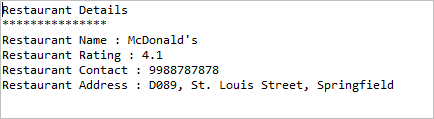
**Method Description**

**Restaurant(String name, long restaurantContact, String restaurantAddress, float rating)**

* Initialize the instance variables appropriately with the values passed to the constructor.

Create an object of the Restaurant class and invoke the displayRestaurantDetails() method in the main() method of the Tester class.

**Sample Output**



**7. What will be the output of the following code?**

public class Tester {

public static void main(String[] args) {

int sum = 0, count = 0;

int[] sales = { 6, 9, 7, 10, 11, 9, 7, 12, 14, 15, 13, 11 };

for (int index = 0; index < sales.length; index++) {

sum += sales[index];

}

float average = (float) sum / sales.length;

for (int sale : sales) {

if (sale > average)

count++;

break;

}

System.out.println("Average sales: " + average);

System.out.println("Sales above average: " + count);

}

}

**8. What will be the output of the below code?**

public class Tester {

public static void main(String[] args) {

int a[][] = { { 1, 3, 4 }, { 2, 3, 6 }, { 7, 6, 5 } };

int sum = 0;

for (int i = 0; i < a.length; i++) {

for (int j = 0; j < a[0].length; j++) {

if (a[i][j] % 2 == 0)

break;

sum += a[i][j];

}

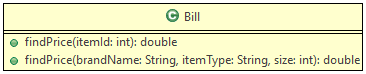
}

System.out.println("sum = " + sum);

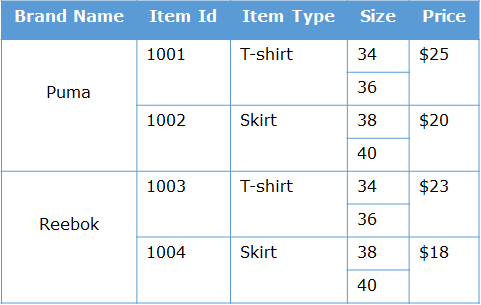
}

}

9. The Bill class is used to find the price of items for calculation. Implement a class Bill based on the class diagram and description given below.



The details of the items are given below.



**Method Description**

**findPrice(int itemId)**

Find and return the price based on the itemId using the table given above.

If the itemId passed to method is invalid, return the price as 0.

**findPrice(String brandName, String itemType, int size)**

Find and return the price based on the brandName, itemType and size using the table given above.

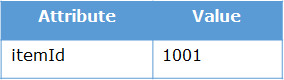
If any invalid details are passed to the method, return the price as 0.

Test the functionalities using the provided Tester class.

**Sample Input and Output**

**For findPrice(int itemId)**

**Input**

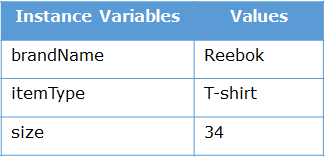


**Output**



**For findPrice(String brandName, String itemType, int size)**

**Input**



**Output**

