## **Week 5**

## Question 1 (1 point)

Saved

A Java constructor is like a method without \_\_\_.

Question 1 options:

|  | statements |
| --- | --- |
|  | **return type** |
|  | argument list |
|  | None |

## Question 2 (1 point)

Saved

The placement of a constructor inside a class should be \_\_\_.

Question 2 options:

|  | Always at the beginning of class |
| --- | --- |
|  | Always at the end of class |
|  | **Anywhere in the class** |
|  | None |

## Question 3 (1 point)

Saved

Memory is allocated to an object once the execution of \_\_\_ is over in Java language.

Question 3 options:

|  | main method |
| --- | --- |
|  | **constructor** |
|  | Destructor |
|  | None |

## Question 4 (1 point)

Saved

Overloading of constructors in Java means adding more than \_\_\_ constructors with the different argument list.

Question 4 options:

|  | **1** |
| --- | --- |
|  | 2 |
|  | 3 |
|  | 4 |

## Question 5 (1 point)

Saved

A constructor can call another overloaded constructor using the \_\_\_ keyword in Java.

Question 5 options:

|  | Super |
| --- | --- |
|  | Local |
|  | **This** |
|  | Con |

## Week 6

## Question 1 (1 point)

Which of these method of String class is used to obtain character at specified index?

Question 1 options:

|  | char() |
| --- | --- |
|  | Charat() |
|  | cht() |
|  | **charAt**() |

## Question 2 (1 point)

Which of these keywords is used to refer to member of base class from a subclass?

Question 2 options:

|  | Upper |
| --- | --- |
|  | **Super** |
|  | this |
|  | String |

## Question 3 (1 point)

Which of the following statements are incorrect?

Question 3 options:

|  | String is a class |
| --- | --- |
|  | **Strings in java are mutable** |
|  | Every string is an object of class String |
|  | Java defines a peer class of String, called StringBuffer, which allows string to be altered |

## Question 4 (1 point)

What will be the output of the following Java program?

1. class string\_class
2. {
3. public static void main(String args[])
4. {
5. String obj = "I LIKE JAVA";
6. System.out.println(obj.length());
7. }
8. }

Question 4 options:

|  | 9 |
| --- | --- |
|  | 10 |
|  | **11** |
|  | 12 |

## Question 5 (1 point)

What will be the output of the following Java program?

1. class string\_class
2. {
3. public static void main(String args[])
4. {
5. String obj = "hello";
6. String obj1 = "world";
7. String obj2 = "hello";
8. System.out.println(obj.equals(obj1) + " " + obj.equals(obj2));
9. }
10. }

Question 5 options:

|  | false false |
| --- | --- |
|  | true true |
|  | true false |
|  | **false true** |

## 

## 

## Week 7

## Question 1 (1 point)

Saved

Which of this keyword must be used to inherit a class?

Question 1 options:

|  | super |
| --- | --- |
|  | **extend** |
|  | extends |
|  | this |

## Question 2 (1 point)

Saved

What will be the output of the following Java program?

1. class A
2. {
3. int i;
4. void display()
5. {
6. System.out.println(i);
7. }
8. }
9. class B extends A
10. {
11. int j;
12. void display()
13. {
14. System.out.println(j);
15. }
16. }
17. class inheritance\_demo
18. {
19. public static void main(String args[])
20. {
21. B obj = new B();
22. obj.i=1;
23. obj.j=2;
24. obj.display();
25. }
26. }

Question 2 options:

|  | 1 |
| --- | --- |
|  | **2** |
|  | 3 |
|  | 0 |

## Question 3 (1 point)

Saved

What will be the output of the following program?

public class Final

{

final int assign;

public static void main(String[] args)

{

final int result = 20;

Final f = new Final();

f.assign = process(result);

System.out.println(f.assign);

}

final static int process(int a)

{

return a + 5;

}

}

Question 3 options:

|  | 20 |
| --- | --- |
|  | 25 |
|  | **Run Time Error** |
|  | Compiler Error |

## Question 4 (1 point)

Saved

If a class inheriting an abstract class does not define all of its function then it will be known as?

Question 4 options:

|  | Abstract |
| --- | --- |
|  | A simple class |
|  | **Static class** |
|  | Interface |

## Question 5 (1 point)

Saved

Which of these is not a correct statement?

Question 5 options:

|  | Every class containing abstract method must be declared abstract |
| --- | --- |
|  | Abstract class defines only the structure of the class not its implementation |
|  | **Abstract class can be initiated by new operator** |
|  | Abstract class can be inherited Week 8Question 1 (1 point)   Saved  Which of these access specifiers must be used for main() method?  Question 1 options:   |  | private | | --- | --- | |  | **public** | |  | protected |  Question 2 (1 point)   Saved  Which of these is used to access a member of class before object of that class is created?  Question 2 options:   |  | public | | --- | --- | |  | private | |  | **static** | |  | protected |  Question 3 (1 point)   Saved  What is the process by which we can control what parts of a program can access the members of a class?  Question 3 options:   |  | Recursion | | --- | --- | |  | Abstraction | |  | **Polymorphism** | |  | Encapsulation |  Question 4 (1 point)   Saved  What will be the output of the following Java code?   1. class access 2. { 3. public int x; 4. private int y; 5. void cal(int a, int b) 6. { 7. x = a + 1; 8. y = b; 9. } 10. } 11. public class access\_specifier 12. { 13. public static void main(String args[]) 14. { 15. access obj = new access(); 16. obj.cal(2, 3); 17. System.out.println(obj.x + " " + obj.y); 18. } 19. }   Question 4 options:   |  | 3 3 | | --- | --- | |  | 2 3 | |  | **Runtime Error** | |  | Compilation Error |  Question 5 (1 point)   Saved  What will be the output of the following Java code?   1. class static\_out 2. { 3. static int x; 4. static int y; 5. void add(int a, int b) 6. { 7. x = a + b; 8. y = x + b; 9. } 10. } 11. public class static\_use 12. { 13. public static void main(String args[]) 14. { 15. static\_out obj1 = new static\_out(); 16. static\_out obj2 = new static\_out(); 17. int a = 2; 18. obj1.add(a, a + 1); 19. obj2.add(5, a); 20. System.out.println(obj1.x + " " + obj2.y); 21. } 22. }   Question 5 options:   |  | **7 9** | | --- | --- | |  | 7 7.4 | |  | 6 6.4 | |  | 9 7 | |

## Week 9

## Question 1 (1 point)

Saved

When does Exceptions in Java arises in code sequence?

Question 1 options:

|  | **Run Time** |
| --- | --- |
|  | Compilation Time |
|  | Can Occur Any Time |
|  | None of the mentioned |

## Question 2 (1 point)

Saved

Which of these keywords must be used to monitor for exceptions?

Question 2 options:

|  | **try** |
| --- | --- |
|  | finally |
|  | catch |
|  | throw |

## Question 3 (1 point)

Saved

Which of these keywords is used to manually throw an exception?

Question 3 options:

|  | try |
| --- | --- |
|  | **throw** |
|  | catch |
|  | finally |

## Question 4 (1 point)

Saved

What will be the output of the following Java program?

1. class exception\_handling
2. {
3. public static void main(String args[])
4. {
5. try
6. {
7. System.out.print("Hello" + " " + 1 / 0);
8. }
9. catch(ArithmeticException e)
10. {
11. System.out.print("World");
12. }
13. }
14. }

Question 4 options:

|  | Hello |
| --- | --- |
|  | **World** |
|  | HelloWorld |
|  | Hello World |

## Question 5 (1 point)

Saved

What will be the output of the following Java program?

1. class exception\_handling
2. {
3. public static void main(String args[])
4. {
5. try
6. {
7. int i, sum;
8. sum = 10;
9. for (i = -1; i < 3 ;++i)
10. sum = (sum / i);
11. }
12. catch(ArithmeticException e)
13. {
14. System.out.print("0");
15. }
16. System.out.print(sum);
17. }
18. }

Question 5 options:

|  | 0 |
| --- | --- |
|  | 5 |
|  | **Compilation Error** |
|  | Runtime Error |

## Week 10

## Question 1 (1 point)

Saved

1. In java a thread can be created by ..........

Question 1 options:

|  | None of these |
| --- | --- |
|  | **Extending the thread class.** |
|  | Both of the above |
|  | Implementing Runnable interface |

## Question 2 (1 point)

Saved

How many threads can be executed at a time?

Question 2 options:

|  | 3 |
| --- | --- |
|  | **1** |
|  | 10 |
|  | 0 |

## Question 3 (1 point)

Saved

Which will contain the body of the thread?

Question 3 options:

|  | main() |
| --- | --- |
|  | stop() |
|  | **run()** |
|  | start() |

## Question 4 (1 point)

Saved

Which of these method of Thread class is used to Suspend a thread for a period of time?

Question 4 options:

|  | sleep() |
| --- | --- |
|  | **suspend()** |
|  | terminate() |
|  | stop() |

## Question 5 (1 point)

Saved

Which function of pre defined class Thread is used to check weather current thread being checked is still running?

Question 5 options:

|  | join() |
| --- | --- |
|  | isRunning() |
|  | Alive() |
|  | **isAlive()** |

## Week 11

## Question 1 (1 point)

Saved

Which of these are legacy classes?

Question 1 options:

|  | Stack |
| --- | --- |
|  | **HashTable** |
|  | Vector |
|  | All the above |

## Question 2 (1 point)

Saved

Which of these is the interface of legacy?

Question 2 options:

|  | **Enumeration** |
| --- | --- |
|  | Map |
|  | HashMap |
|  | Vector |

## Question 3 (1 point)

Saved

Which of these methods is used to add elements in vector at specific location?

Question 3 options:

|  | add() |
| --- | --- |
|  | **addElement**() |
|  | AddElement() |
|  | set() |

## Question 4 (1 point)

Saved

1. import java.util.\*;
2. class vector
3. {
4. public static void main(String args[])
5. {
6. Vector obj = new Vector(4,2);
7. obj.addElement(new Integer(3));
8. obj.addElement(new Integer(2));
9. obj.addElement(new Integer(5));
10. System.out.println(obj.elementAt(1));
11. }
12. }

Question 4 options:

|  | 0 |
| --- | --- |
|  | **2** |
|  | 3 |
|  | 5 |

## Question 5 (1 point)

Saved

1. import java.util.\*;
2. class stack
3. {
4. public static void main(String args[])
5. {
6. Stack obj = new Stack();
7. obj.push(new Integer(3));
8. obj.push(new Integer(2));
9. obj.pop();
10. obj.push(new Integer(5));
11. System.out.println(obj);
12. }
13. }

Question 5 options:

|  | **[3,5]** |
| --- | --- |
|  | [3,2] |
|  | [3,2,5] |
|  | [3,5,2] |

## Week 12

## Question 1 (1 point)

Saved

Which string method eliminates whitespaces before and after tokens?

Question 1 options:

|  | join() |
| --- | --- |
|  | intern( ) |
|  | format( ) |
|  | **trim**( ) |

## Question 2 (1 point)

Saved

How to format date from one form to another?

Question 2 options:

|  | **SimpleDateFormat** |
| --- | --- |
|  | DateFormat |
|  | SimpleFormat |
|  | DateConverter |

## Question 3 (1 point)

Saved

How to get difference between two dates?

Question 3 options:

|  | **long diffInMilli = java.time.Duration.between(dateTime1, dateTime2).toMillis();** |
| --- | --- |
|  | long diffInMilli = java.time.difference(dateTime1, dateTime2).toMillis(); |
|  | Date diffInMilli = java.time.Duration.between(dateTime1, dateTime2).toMillis(); |
|  | Time diffInMilli = java.time.Duration.between(dateTime1, dateTime2).toMillis(); |

## Question 4 (1 point)

Saved

Which of these methods is called when observed object has changed?

Question 4 options:

|  | setChanged() |
| --- | --- |
|  | **update()** |
|  | notifyObserver() |
|  | all of the mentioned |

## Question 5 (1 point)

Saved

Which of these package provides the ability to read and write in Zip format?

Question 5 options:

|  | java.lang |
| --- | --- |
|  | java.io |
|  | **java.util.zip** |
|  | java.util.zip |