**Question 1**

What is the output for the following code?

**public** **class** TestSet **extends** Thread {

**public** **static** **void** main(String argv[]) {

TestSet b = **new** TestSet();

b.start();

}

**public** **void** run() {

System.*out*.println("Running");

}

}

**Answer:** Compilation and run with the output "Running"

**Question 2**

What is the result of attempting to compile and run the following code?

**public** **interface** TestInf {

**int** *i* = 10;

}

**public** **class** Test {

**public** **static** **void** main(String[] args) {

TestInf.*i* = 12;

System.*out*.println(TestInf.*i*);

}

}

**Answer:** Compile with error.

Explanations : All the variables declared in interface is implicitly static and final , therefore can't change the value.

**Question 3**

What is the result of executing the following code, using the parameters 0 and 3 (a=0, b=3)?

**public** **void** divide(**int** a, **int** b) {

**try** {

**int** c = a / b;

} **catch** (Exception e) {

System.*out*.print("Exception ");

} **finally** {

System.*out*.println("Finally");

}

**Answer:** Prints out: Finally

Explanations : finally block always executed whether exception occurs or not.

0/3 = 0 Does not throws exception.

**Question 4**

What are synchronized methods and synchronized statements?

**Answer:** Synchronized methods are methods that are used to control access to an object. A thread only executes a synchronized method after it has acquired the lock for the method's object or class. Synchronized statements are similar to synchronized methods. A synchronized statement can only be executed after a thread has acquired the lock for the object or class referenced in the synchronized statement.

**Question 5**

What is the output for the following code running in the same JVM?

**public** **class** A **implements** Serializable {

**transient** **int** a = 7;

**static** **int** *b* = 9;

}

**public** **class** B **implements** Serializable {

**public** **static** **void** main(String[] args) {

A a = **new** A();

**try** {

ObjectOutputStream os = **new** ObjectOutputStream(

**new** FileOutputStream("test.ser"));

os.writeObject(a);

os.close();

System.*out*.print(+ +a.*b* + " ");

ObjectInputStream is = **new** ObjectInputStream(**new** FileInputStream(

"test.ser"));

A s2 = (A) is.readObject();

is.close();

System.*out*.println(s2.a + " " + s2.*b*);

} **catch** (Exception x) {

x.printStackTrace();

}

}

}

**Anwer:** 9 0 9

**Question 6**

Can an abstract method have the static qualifier?

**Anwer:** No

**Question 7**

What is early binding and late binding?

**Anwer:**

Connecting a method call(i.e. Function Call) to a method body(i.e. Function) is called binding. When binding is performed before the program is run (by the compiler and linker, if there is one), it’s called early binding. You might not have heard the term before because it has never been an option with procedural languages. C compilers have only one kind of method call, and that’s early binding. The other solution is called late binding, which means that the binding occurs at run time, based on the type of object. Late binding is also called dynamic binding or run-time binding.

**Question 8**

Can you call one constructor from another if a class has multiple constructors?

**Anwer:** Yes, using this()

**Question 9**

What's the difference between the methods sleep() and wait()?

**Anwer:**

The static Thread.sleep(long) method maintains control of thread execution but delays the next action until the sleep time expires. The wait method gives up control over thread execution indefinitely so that other threads can run.

**Question 10**

You are planning to do an indexed search in a list of objects. Which of the two Java collections should you use: ArrayList or LinkedList?

**Anwer:** ArrayList

**Question 11**

How do I include static files within a JSP page?

**Anwer:**

Using include directive <%@ include file="copyright.html" %>

**Question 12**

What is the page directive that is used to prevent a JSP page from automatically creating a session?

**Anwer:** <%@ page session="false">

**Question 13**

What is the difference between ServletContext and PageContext?

**Anwer:**

The ServletContext gives information about the container in which the servlet (or JSP) is running in. Parameters for this can be setup in the web application deployment descriptor and there is one ServletContext per web application.

The PageContext gives the servlet (or JSP) information about the request that it is currently handling and contains information about the request and any parameters, the session, the response object, a reference to the output stream and also a reference to the web application's ServletContext.

**Question 14**

What is the difference in using request.getRequestDispatcher() and context.getRequestDispatcher()?

**Anwer:**

The servletRequest's getRequestDispatcher() can take a relative path while  
ServletContext's getRequestDispatcher() can not (can only take relative to the   
current context's root).

In request.getRequestDispatcher(path) in order to create it we need to give the relative path of the resource. But in resourcecontext.getRequestDispatcher(path) in order to create it we need to give the absolute path of the resource.

**Question 15**

What is the difference between ServletContext and ServletConfig?

**Anwer:**

ServletContext defines a set of methods that a servlet uses to communicate with its servlet container.   
ServletConfig is a servlet configuration object used by a servlet container used to pass information to a servlet during initialization. All of its initialization parameters can ONLY be set in deployment descriptor.

The ServletContext object is contained within the ServletConfig object, which the Web server provides the servlet when the servlet is initialized.

You can specify param-value pairs for ServletContext object in <context-param> tags in web.xml file.

The ServletConfig parameters are specified for a particular servlet and are unknown to other servlets.

The ServletContext parameters are specified for an entire application outside of any particular servlet and are available to all the servlets within that application

**Question 16**

Explain bubble sort with code snippet and what is its complexity?

O(n2)

public **void** bubbleSort(**int**[] arr) {

**boolean** swapped = **true**;

**int** j = 0;

**int** tmp;

**while** (swapped) {

            swapped = **false**;

            j++;

**for** (**int** i = 0; i < arr.length - j; i++) {

**if** (arr[i] > arr[i + 1]) {

                        tmp = arr[i];

                        arr[i] = arr[i + 1];

                        arr[i + 1] = tmp;

                        swapped = **true**;

                  }

            }

      }

}

**Question 17**

Write an algorithm to find the factorial of a given number?

public static long factorial(long n) {

if (n <= 1)

return 1;

else

return n\*factorial(n-1);

}

**Question 18**

Write an algorithm to find the number of occurrences of an integer in a given array?

/\*\* Counts the number of times an integer appears in an array. \*/

public static int findCount(int[] a, int k) {

int count = 0;

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

if (a[i]== k) // check if the current element equals k

count++;

}

return count;

}

**Question 19**

Write the iterative linear algorithm to get the Fibonacci series?

**public** **class** FibonacciIterative **{**

**public** **static** **int** fib**(int** n**)** **{**

**int** prev1=0, prev2=1;

**for(int** i=0; i<n; i++**)** **{**

**int** savePrev1 = prev1;

prev1 = prev2;

prev2 = savePrev1 + prev2;

**}**

**return** prev1;

**}**

**}**

**Question 20**

Write an algorithm to illustrate Merge Sort?

void mergesort(int lo, int hi)

{

if (lo<hi)

{

int m=(lo+hi)/2;

mergesort(lo, m);

mergesort(m+1, hi);

merge(lo, m, hi);

}

}

// Straightforward variant

void merge(int lo, int m, int hi)

{

int i, j, k;

// copy both halves of a to auxiliary array b

for (i=lo; i<=hi; i++)

b[i]=a[i];

i=lo; j=m+1; k=lo;

// copy back next-greatest element at each time

while (i<=m && j<=hi)

if (b[i]<=b[j])

a[k++]=b[i++];

else

a[k++]=b[j++];

// copy back remaining elements of first half (if any)

while (i<=m)

a[k++]=b[i++];

}