1. **Explain the differences between primitive and reference data types.**
   The differences between primitive and reference data type is that reference data can be stored as null while primitive cannot.

2. **Define the scope of a variable (hint: local and global variable)**
   Scope of a variable is the part of the program where the variable is accessible.

3. **Why is initialization of variables required.**
   Variables should be initialized are the compiler return an error.

4. **Differentiate between static, instance and local variables.**
   Local variables exist inside a block of a method and only exist if the method is being executed. Instance variables is associated with the class in which it is defined. Static variables are associated with the whole class.

5. **Differentiate between widening and narrowing casting in java.**
   Widening is assigning smaller type to larger type while narrowing is assigning a larger type value to smaller type.

6. **the following table shows data type, its size, default value and the range. Filling in the missing values.**

TYPE	SIZE (IN BYTES)	DEFAULT	RANGE
boolean	1 bit		true, false
Char	2		'\0000' to '\ffff'
Byte		0	-2^{7} to $+2^{7}-1$
Short		0	-2^{15} to $+2^{15}-1$
Int	4		-2^{31} to $+2^{31}-1$
Long		0L	-
Float	4	00.0f	
Double	8		-1.8E+308 to +1.8E+308

7.Explain the importance of using Java packages

a) Implementing Data Encapsulation in Java is easy with the help of packages.
b) Packages help prevent naming conflicts.
c) It orders the classes according to their functions. Hence it is easy to search for classes.
d) You can control the accessibility of the classes by using access specifiers and packages together. A detailed table of access controls in packages is present at the end of the article.

8.Explain three controls used when creating GUI applications in Java language.

a) Text
b) Link
c) Picture

9.Explain the difference between containers and components as used in Java.

A component is an object with a visual representation while a container is a subclass of a component which can contain other components.

10.Write a Java program to reverse an array having five items of type int.

```
   Public class ReverseArray{

     Public static void main(String[] args)
      //initialize array
    Int [] arr new int []{1,2,3,4,5};
  System.out.println("original array":);
 For(int i=0; i<arr.length;++){
 System.out.print(arr[i]+**);
}
  System.out.println()
 System.out.println("array in reverse order")
 //loop through the array in reverse order
For(int i=arr.length-1;i>=0;i--){
System.out.print(arr[i]+"""");
      }
   }
}
```

11.**Programs written for a graphical user interface have to deal with "events."**

Explain what is meant by the term event.

An event is an object created when something changes within a graphical user interface.

Give at least two different examples of events, and discuss how a program might respond to those events.

a) Background events. Require user interactions to generate.
b) Foreground events. Don't require any user interaction.

12.**Explain the difference between the following terms as used in Java programming.**

Polymorphism and encapsulation

Polymorphism is the ability of a message to be displayed in more than one form while encapsulation is binding code and its data.

method overloading and method overriding

Method overloading is used to increase the readability of a program while method overriding is used to provide specific implementation of the method that is already in a class.

class and interface

A class is where objects are created while an interface specifies what a class must do and when .

inheritance and polymorphism

Inheritance is the constant relationship between classes while polymoriphism is processing objects differently based on their data type.

13.sing examples, explain the two possible ways of implementing polymorphism. Show your code in java.

1. Static or compile-time polymorphism is done by method overloading

2. Dynamic or runtime polymorphism is done by method overriding

1. With relevant examples, explain the following concepts as used in Java programming.

a. Mutable classes.

Explain what is meant by mutable class

A mutable class is a class with objects whose value can be changed.

Write a program that implements the concept of mutable class

```
            public class JtpExample {
        private String s;
        JtpExample(String s) {
        this.s = s;
  }
  public String getName() {
        return s;
  }
  public void setName(String coursename) {
  this.s = coursename;
  }
  public static void main(String[] args) {
  JtpExample obj = new JtpExample("JavaTpoint");
  System.out.println(obj.getName());
// Here, we can update the name using the setName method.
  obj.setName("Java Training");
  System.out.println(obj.getName());
}
}
```

b. Immutable classes.

Explain what is meant by immutable class

An immutable class is a class that has objects whose value that cannot be changed.

Write a program that implements the concept of immutable class

```
        public class JtpExample1 {
      private final String s;
      JtpExample1(final String s) {
      this.s = s;
      }
      public final String getName() {
      return s;
      }
      public static void main(String[] args) {
      JtpExample obj = new JtpExample("Core Java Training");
      System.out.println(obj.getName());
      }
     }
```

c. Explain the situations where mutable classes are more preferable than immutable classes when writing a Java program.

2. Explain what a String buffer class is as used in Java

It is used to create mutable string.

b. Write the output of the following program.

Hello I love java

```
class Myoutput
{
      public static void main(String args[])
      {
            String ast = "hello i love java";
            System.out.println(ast.indexOf('e')+"
"+ast.indexOf('ast')+" "+ast.lastIndexOf('l')+"
"+ast .lastIndexOf('v'));
      }
}
```

c. Explain your answer in (2b) above.

String ast= “hello I love java”

d. With explanation, write the output of the following program.

Jambo Kenya

```
class Myoutput
{
      public static void main(String args[])
      {
             StringBuffer bfobj = new StringBuffer("Jambo");
             StringBuffer bfobj1 = new StringBuffer(" Kenya");
             c.append(bfobj1);
             System.out.println(bfobj);
      }
}
```

e. With explanation, write the output of the following program.

Jambo

```
class Myoutput
        {
            public static void main(String args[])
            {
               StringBuffer str1 = new StringBuffer("Jambo");
               StringBuffer str2 = str1.reverse();
               System.out.println(str2);
            }
        }
```

f. With explanation, write the output of the following program.

```
class Myoutput
        {
           class output
           {
             public static void main(String args[])
             {
                char c[]={'A', '1', 'b' ,' ' ,'a' , '0'};
                for (int i = 0; i < 5; ++i)
                {
                       i++;
                             if(Character.isDigit(c[i]))
                                 System.out.println(c[i]+" is a digit");
                             if(Character.isWhitespace(c[i]))
                                 System.out.println(c[i]+" is a Whitespace character");
```

```
            if(Character.isUpperCase(c[i]))
                System.out.println(c[i]+" is an Upper case Letter");
            if(Character.isLowerCase(c[i]))
                System.out.println(c[i]+" is a lower case Letter");
            i++;
        }
    }
}
```