Q>

```
public class ExampleDoWhile{
      public static void main(String args[]){
                  do{
                        System.out.println("hello");//line-n1
                  }while(true);
                  System.out.println("hi");//line-n2
      }
}
```

A. CompileTime Error at line-n1
B. hello infinite times
C. hi
D. some problem by jvm during the execution
E. CompileTime Error at line-n2
F. None of the above

Answer: E

Q>

```
public class ExampleDoWhile{
      public static void main(String args[]){
                  do{
                        System.out.println("hello");//line-n1
                  }while(false);
                  System.out.println("hi");//line-n2
      }
}
```

A. CompileTime Error at line-n1
B. hello
C. hi
D. some problem by jvm during the execution
E. CompileTime Error at line-n2
F. hello
    hi
G. None of the above

Answer: F

Q>

```
public class ExampleDoWhile{
            public static void main(String args[]){
                        int a=10,b=20;
                        do{
                              System.out.println("hello");//line-n1
                        } while(a<b);// JVM ----> while(10<20) ---> while(true)
                        System.out.println("hi");//line-n2
         }
}
```

A. CompileTime Error at line-n1
B. hello
C. hi
D. some problem by jvm during the execution
E. CompileTime Error at line-n2
F. hello infinite times
g. hi infinite times
h. None of the above

Answer: F

Q>

```
public class ExampleDoWhile{
            public static void main(String args[]){
                        int a=10,b=20;
                        do{
                              System.out.println("hello");//line-n1
                        } while(a>b);//JVM ---> while(10>20) ----> while(false)
                        System.out.println("hi");//line-n2
         }
}
A. CompileTime Error at line-n1
B. hello
     hi
C. hi
     hello
D. some problem by jvm during the execution
E. CompileTime Error at line-n2
F. hello infinite times
    hi
G. None of the above
```

Answer: B

Q>

```
public class ExampleDoWhile{
            public static void main(String args[]){
                        final int a=10,b=20;
                        do{
                              System.out.println("hello");//line-n1
                        } while(a<b);//Compiler-----> while(10<20) ---> while(true)
                        System.out.println("hi");//line-n2
         }
}
A. CompileTime Error at line-n1
B. hello
     hi
C. hi
     hello
D. some problem by jvm during the execution
E. CompileTime Error at line-n2
F. hello infinite times
    hi
G. None of the above
```

Answer: E(concept of unreachable)

Q>

```
public class ExampleDoWhile{
            public static void main(String args[]){
                        final int a=10,b=20;
                        do{
                              System.out.println("hello");//line-n1
                        } while(a>b);//Compiler ----> while(10>20) ---->
while(false)
                        System.out.println("hi");//line-n2
         }
}
A. CompileTime Error at line-n1
B. hello
```

```
     hi
C. hi
     hello
D. some problem by jvm during the execution
E. CompileTime Error at line-n2
F. hello infinite times
    hi
G. None of the above
```

Answer: B

Q>

```
int i=0,j=0; //line -n1
int i=0,Boolean b=true; //line-n2
int i=0,int j=0; //line -n3
```

How many statements are valid?

```
 A. line -n1 and line -n3
 B. line -n2
 C. line-n1, line-n2 and line-n3
 D.line -n3
 E. line -n1
```

Answer: E(after , in declartion we need to just specify the variables only)

Q>

Syntax:

```
    for(stmt1;stmt2;stmt3){
            stmt4;
    }
      stmt1 -> can be any statement,but suggested for intitalisation
      stmt2 -> compulsorily should be boolean statement only
      stm3  -> can be any statement, but suggested for inc/dec on a variable
      stm4  -> can be any statement,suggested for repetative logic
```

```
public class ExampleFor{
            public static void main(String args[]){
                        int i=0;
                        for(System.out.println("hello u r sleeping");i<3;i++){
                                    System.out.println("no boss, u only sleeping");
                        }
            }
}
```

Predict the Output:

```
A.Compile Time Error
B. Some problem occured by jvm during execution
C. hello u r sleeping
D. no boss, u only sleeping
E.
hello u r sleeping
No boss, u only sleeping
No boss, u only sleeping
No boss, u only sleeping
```

Answer: E

Q>

```
public class ExampleFor{
      public static void main(String args[]){
            int i=0;
            for(System.out.println("hello");i<3;System.out.println("hi")){
                        i++;
            }
      }
}
Predict the Output:
A.Compile Time Error
B. Some problem occured by jvm during execution
C. hello
     hi
D.hello
    hi
    hi
    hi
E. hi
    hi
    hi
   hello
F. None of the above
```

Answer: D

```
Q>
public class ExampleFor{
            public static void main(String args[]){
                  for(;;){//Compiler---> boolean value will be evaulated as 'true'
                        System.out.println("hello");
                  }
            }
}
Predict the Output:
A.Compile Time Error
B. Some problem occured by jvm during execution
C. hello
D. infinite times hello
E. None of the above
```

Answer: D

```
Q>
public class ExampleFor{
      public static void main(String args[]){
                  for(int i=0;true;i++){
                              System.out.println("hello");//line-n1
                  }
                  System.out.println("hi");//line-n2
      }
}
Predict the Output:
A.Compile Time Error at line-n1
B.Compile Time Error at line-n2
C. Some problem occured by jvm during execution
D. hello
     hi
D. infinite times hello followed by hi
E. None of the above
```

Answer: B

Q>

```
public class ExampleFor{
      public static void main(String args[]){
                  for(int i=0;false;i++){
                              System.out.println("hello");//line-n1
                  }
                  System.out.println("hi");//line-n2
      }
}
```

Predict the Output:
A.Compile Time Error at line-n1
B.Compile Time Error at line-n2
C. Some problem occured by jvm during execution
D. hello
     hi
D. infinite times hello followed by hi
E. None of the above

Answer: A(remember the concepts of unreachable)

Q>

```
public class ExampleFor{
            public static void main(String args[]){
                  for(int i=0;;i++){
                              System.out.println("hello");//line-n1
                  }
                  System.out.println("hi");//line-n2
            }
}
```

Predict the Output:
A.Compile Time Error at line-n1
B.Compile Time Error at line-n2
C. Some problem occured by jvm during execution
D. hello
     hi
D. infinite times hello
E. None of the above

Answer : B

Q>

```
public class ExampleFor{
      public static void main(String args[]){
            int a=10,b=20;
            for(int i=0;a<b;i++){//JVM ----> 10<20 ( true)
                        System.out.println("hello");//line-n1
            }
            System.out.println("hi");//line-n2
      }
}
```

Predict the Output:
A.Compile Time Error at line-n1
B.Compile Time Error at line-n2
C. Some problem occured by jvm during execution
D. hello
     hi
E. infinite times hello

F. None of the above

Answer: E

Q>

```
public class ExampleFor{
            public static void main(String args[]){
                        final int a=10,b=20;
                        for(int i=0;a<b;i++){//Compiler ---->  10<20 (true)
                                    System.out.println("hello");//line-n1
                        }
                        System.out.println("hi");//line-n2
            }
}
```

Predict the Output:
A.Compile Time Error at line-n1
B.Compile Time Error at line-n2
C. Some problem occured by jvm during execution
D. hello
     hi
E. infinite times hello
F. None of the above

Answer: B

Note: if a variable is marked as final, then those values are known to compiler so we say final variables as "CompiletimeConstants".

if a variable is marked as final, then the value for those variables should never be changed in the program, if we try to change it would result in "CompileTimeError".

In java memory for a variable is given by JVM as per its datatype specification and value also will be assigned.

compiler will not allocate memory for the variables and it will not initialize the value for the variable.

```
eg: int a =10;
      a++;
      System.out.println(a);//11
```

```
eg: final int a =10;
      a++;//a  = a +1;//CE: value can't be re-assigned
      System.out.println(a);
```