transfer statements
==========
1. break
2. continue
3. return

Q>
```
class Test{
      public static void main(String args[]){
            int x=0;
            switch(x){
                  case 0:System.out.println("hello");
                              break ;//It is used to avoid fallthrough in switch
                  case 1:System.out.println("hi");
            }
      }
}
```
Predict the Output:
A.Compile Time Error
B. Some problem occured by jvm during execution
C.hello
    hi
D. hello

Answer: D

Q>
```
 class Test{
            public static void main(String args[]){
                        for(int i=0; i<10; i++) {// i = 0,0<10(true),i =
1,1<10(true)
                                    if(i==5)
                                     break;//control will come out of the executing
loop
                              System.out.print(i);//0,1,2,3,4
                        }
            }
}
```
Predict the Output:
A.Compile Time Error
B. Some problem occured by jvm during execution
C.0 1 2 3 4
D. 0 1 2 3 4 6 7 8 9

Answer: C

Q>
```
class Test{
            public static void main(String args[]){
                        int x=10;
                        l1 : {
                                    System.out.println("begin");
                                    if(x==10)
                                    break l1;
                                    System.out.println("end");
                              }
                              System.out.println("hello");
```

```
                        }
}
```

Predict the Output:
A.Compile Time Error
B. Some problem occured by jvm during execution
C. begin
     end
D. begin
     end
     hello
E. None of the above

Answer: E(begin and hello will be printed)

Q>

```
class Test{
      public static void main(String args[]){
            int x=10;
            if(x==10)
            break;
            System.out.println("hello");
      }
}
```

Predict the Output:
A.Compile Time Error
B. Some problem occured by jvm during execution
C. hello
D. None of the above

Answer: A (break can be used in switch,loop and labelled block,otherplace compile time error)

Q>

```
class Test{
      public static void main(String args[]){
            int x=2;
            for(int i = 0; i<10;i++) {
                  if(i%x==0)
                   continue;//it will skip the current iteration and proceeds with next iteration
                  System.out.println(i);
            }
      }
}
```

Predict the Output:
A. Compile Time Error
B. Some problem occured by jvm during execution
C. 0 2 4 6 8
D. 1 3 5 7 9
E. None of the above

Answer: D

Q>

```
class Test
{
            public static void main(String args[]){
```

```
                int x=10;
                if(x==10)
                continue;//continue can't be used here
                System.out.println("hello");
          }
}
```
Predict the Output:
A. Compile Time Error
B. Some problem occured by jvm during execution
C. hello
E. None of the above

Answer: A

Q>
```
class Test{
      public static void main(String args[]){
            int x=0;
            switch(x){
                  case 0:System.out.println("hello");
                              continue;
                  case 1:System.out.println("hi");
            }
      }
}
```
Predict the Output:
A.Compile Time Error
B. Some problem occured by jvm during execution
C.hello
    hi
D. hello

Answer: A(continue can be used only in loops and labelled block,otherplaces compile time error)

Note:
```
l1:
for()
{
      l2: for()
      {
       l3: for()
            {

                  break/break l3; // goto stmt-1
                  break l2; //goto stmt2
                  break l1;// goto stmt3
            }
            stmt-1;
      }
            stmt-2;
}
            stmt-3;
```

Q>
```
class Test{
            public static void main(String args[]){
```

```
                  l1:for(int i=0;i<3;i++) // i = 0 ,0<3(true) , i =1, 1<3(true), i
=2 , 2<3(true), i = 3 , 3<3(false)
                              for(int j=0;j<3;j++)// j =0,0<1(true), j =1,
1<3(true),j =2 ,2<3(true)
                              {
                                          if(i==j)
                                                stmt1;

                                          System.out.println(i+""+j);//10 20 21
                              }
                        }
            }
}
```

replace stmt1 with break and predict the output?
Answer:1 0  2 0 2 1

replace stmt1 with break l1 and predict the output?
Answer:  no output

replace stmt1 with continue and predict the output?
Answer:  01, 02,10,12,20,21

replace stmt1 with continue l1 and predict the output?
Answer: 10 20 21

Q>

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

Predict the output
A. Compile time error at line-n1
B. some problem occured during jvm execution.
C. Compile time error at line-n2
D. hello
     hi
E. hello infinite times
F. None of the above

Answer: C

Q>

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

Predict the output
A. compile time error at line-n1

B. some problem occured during jvm execution.
C.Compile time error at line-n2
D. hello
     hi
E. hello
F. hi
G. None of the above

Answer: E

Concept of unreachable holds good only for loops(for,while,dowhile), compiler will ignore unreachable for
if else and switch syntax.

Q>

```
class A
{
   void add(byte a, byte b)
   {
      //stmt;
   }
}

class Test
{
  public static void main(String[] arg)
  {
      A obj=new A();
      obj.add(10,20); //Line 1 => CE: Why it is throwing CE. The value 10 is inbetween byte range(-128 to 127 )

      int i1=10;
      byte b1=10;  // Line 2 => Here Compiler accepting the value 10.
      byte b2=i1;  //Line 3 => CE
  }
}
```