UNIT IX : NESTED LOOP

Types of Nested Loop

- Nested for loop
- Nested while loop
- Nested do-while loop

Name	Nested for loop	Nested while loop	Nested do-while loop
Define	A for loop inside another for loop	A while loop inside another while loop	A do-while loop inside another do-while loop
Example	`for (int i = 0; i < 3; i++) { for (int j = 0; j < 3; j++) { System.out.print("* "); } System.out.println(); }`	`int i = 0; while (i < 3) { int j = 0; while (j < 3) { System.out.print("* "); j++; } System.out.println(); i++; }`	`int i = 0; do { int j = 0; do { System.out.print("* "); j++; } while (j < 3); System.out.println(); i++; } while (i < 3);`
Explanation of example	The nested for loop prints a square of asterisks by printing out three asterisks in each row, and three rows in total.	The nested while loop prints the same square of asterisks by using while loops instead of for loops. The outer loop sets the row counter, and the inner loop sets the column counter.	The nested do-while loop is similar to the nested while loop, but it guarantees that the inner loop will run at least once, even if the outer loop's condition is not met. In this example, it prints the same square of asterisks as the other two examples.

Break and Continue

Feature	break	continue
Function	Terminates the enclosing loop prematurely	Skips the current iteration of the loop and moves on to the next iteration
Execution	Stops the loop execution entirely	Jumps to the next iteration of the loop
Scope	Can be used with any loop construct (for loop, while loop, do-while loop)	Can only be used with loop constructs
Control	Provides complete control over the loop execution	Provides partial control over the loop execution
Effect on Code	Code execution continues after the loop	Code execution continues within the loop

Break statement in a nested loop

(i) Use of break statement in the Outer loop

```
int i,j;
for (i=1; i<=5; i++)
 { for (j=1; j<=3; j++)
    { System.out.println (i*j);
    }
  if (i>2)
     break;
  System.out.println ("* * * *");
 }
```

→ Exits the Outer loop

(ii) Use of break statement in the inner loop

```
int i,j;
 for (i=1; i<=5; i++)
  { for (j=1; j<=3; j++)
     { System.out.println(i*j);
       if (i*j >8)
           break;
     }
  }
```

→ Exits the inner loop

(iii) Labelled Break statement

Sometimes, it may happen that the requirement of the user is fulfilled during the iteration of the inner loop and so the statements in outer loop should not be executed further. Therefore Labelled Break statement is used for the termination of Outer loop from inner loop.

```
int i,j;
outer:
    for(i=1; i<=5; i++)
        {
        for(j=1; j<=3; j++)
            { System.out.println(i*j);
              if (j==2)
                break outer;
        }
}
```

Exits the Outer loop

Similary We can use Continue statement in inner loop, outer loop and labelled continue