**JAVA NOTES FOR UNCOMMON CONCEPTS:**

1. **The for-each loop** is used to traverse array or collection in Java. It is easier to use than simple for loop because we don't need to increment value and use subscript notation.  
   It works on the basis of elements and not the index. It returns element one by one in the defined variable.  
     
   //Java For-each loop example which prints the   
   //elements of the array   
   public class ForEachExample {   
    public static void main(String[] args) {   
    //Declaring an array   
    int arr[]={12,23,44,56,78};   
    //Printing array using for-each loop  **for(int i:arr){** System.out.println(i);   
    }   
    }   
   }   
     
     
   //A Java program to demonstrate the use **of labeled for loop**   
   public class LabeledForExample {   
   public static void main(String[] args) {   
    //Using Label for outer and for loop   
    aa:   
    for(int i=1;i<=3;i++){   
    bb:   
    for(int j=1;j<=3;j++){   
    if(i==2&&j==2){   
    break aa;   
    }   
    System.out.println(i+" "+j);   
    }   
    }   
   }   
   }   
   Output:  
     
   1 1  
   1 2  
   1 3  
   2 1  
   If you use break bb;, it will break inner loop only which is the default behaviour of any loop.  
     
   LabeledForExample2.java  
     
   public class LabeledForExample2 {   
   public static void main(String[] args) {   
    aa:   
    for(int i=1;i<=3;i++){   
    bb:   
    for(int j=1;j<=3;j++){   
    if(i==2&&j==2){   
    break bb;   
    }   
    System.out.println(i+" "+j);   
    }   
    }   
   }   
   }   
   Output:  
     
   1 1  
   1 2  
   1 3  
   2 1  
   3 1  
   3 2  
   3 3  
     
   **Do-while syntax**  
     
   do{   
   //code to be executed / loop body   
   //update statement   
   }while (condition);  
     
   **While syntax**  
   while (condition){   
   //code to be executed   
   Increment / decrement statement   
   }   
     
   **Switch syntax**  
   public class SwitchExample {   
   public static void main(String[] args) {   
    //Declaring a variable for switch expression   
    int number=20;   
    //Switch expression   
    switch(number){   
    //Case statements   
    case 10: System.out.println("10");   
    break;   
    case 20: System.out.println("20");   
    break;   
    case 30: System.out.println("30");   
    break;   
    //Default case statement   
    default:System.out.println("Not in 10, 20 or 30");   
    }   
   }   
   }   
     
   **Break statements**When a break statement is encountered inside a loop, the loop is immediately terminated and the program control resumes at the next statement following the loop.  
     
   The Java break statement is used to break loop or switch statement. It breaks the current flow of the program at specified condition. In case of inner loop, it breaks only inner loop.  
     
   We can use Java break statement in all types of loops such as for loop, while loop and do-while loop.  
     
     
   //Java Program to illustrate the use of continue statement   
   //with label inside an inner loop to break outer loop   
   public class BreakExample3 {   
   public static void main(String[] args) {   
    aa:   
    for(int i=1;i<=3;i++){   
    bb:   
    for(int j=1;j<=3;j++){   
    if(i==2&&j==2){   
    //using break statement with label   
    break aa;   
    }   
    System.out.println(i+" "+j);   
    }   
    }   
   }   
   }   
     
     
   **Continue statement**  
     
   The continue statement is used in loop control structure when you need to jump to the next iteration of the loop immediately. It can be used with for loop or while loop.  
     
   The Java continue statement is used to continue the loop. It continues the current flow of the program and skips the remaining code at the specified condition. In case of an inner loop, it continues the inner loop only.  
     
   We can use Java continue statement in all types of loops such as for loop, while loop and do-while loop.  
     
     
   //Java Program to demonstrate the use of continue statement   
   //inside the while loop.   
   public class ContinueWhileExample {   
   public static void main(String[] args) {   
    //while loop   
    int i=1;   
    while(i<=10){   
    if(i==5){   
    //using continue statement   
    i++;   
    continue;//it will skip the rest statement   
    }   
    System.out.println(i);   
    i++;   
    }   
   }   
   }  
     
   Output:  
   1  
   2  
   3  
   4  
   6  
   7  
   8  
   9  
   10  
     
   **Java documentation comment**Documentation comments are usually used to write large programs for a project or software application as it helps to create documentation API. These APIs are needed for reference, i.e., which classes, methods, arguments, etc., are used in the code.  
     
   To create documentation API, we need to use the javadoc tool. The documentation comments are placed between /\*\* and \*/.  
     
   **syntax:**  
   /\*\*   
   \*   
   \*We can use various tags to depict the parameter   
   \*or heading or author name   
   \*We can also use HTML tags   
   \*   
   \*/   
   **Javadoc tags**Some of the commonly used tags in documentation comments:  
     
     
     
   **Calculate.java**  
     
   import java.io.\*;   
      
   /\*\*   
    \* <h2> Calculation of numbers </h2>   
    \* This program implements an application   
    \* to perform operation such as addition of numbers   
    \* and print the result   
    \* <p>   
    \* <b>Note:</b> Comments make the code readable and   
    \* easy to understand.   
    \*   
    \* @author Anurati   
    \* @version 16.0   
    \* @since 2021-07-06   
    \*/   
      
    public class Calculate{   
    /\*\*   
    \* This method calculates the summation of two integers.   
    \* @param input1 This is the first parameter to sum() method   
    \* @param input2 This is the second parameter to the sum() method.   
    \* @return int This returns the addition of input1 and input2   
    \*/   
    public int sum(int input1, int input2){   
    return input1 + input2;   
    }   
    /\*\*   
    \* This is the main method uses of sum() method.   
    \* @param args Unused   
    \* @see IOException   
    \*/   
    public static void main(String[] args) {   
    Calculate obj = new Calculate();   
    int result = obj.sum(40, 20);   
      
    System.out.println("Addition of numbers: " + result);   
    }   
    }   
   Compile it by javac tool:  
     
   Create Document  
     
     
   Create documentation API by javadoc tool:  
     
     
     
   Now, the HTML files are created for the Calculate class in the current directory, i.e., abcDemo. Open the HTML files, and we can see the explanation of Calculate class provided through the documentation comment.  
     
     
     
   
2. a
3. a
4. a
5. a
6. a
7. a
8. a
9. a
10. a
11. a
12. a
13. a
14. a
15. a
16. a
17. a
18. a
19. a
20. a
21. a
22. a
23. a
24. a
25. a
26. a
27. a
28. a
29. a
30. a
31. a
32. a
33. a
34. a
35. a
36. a
37. a
38. a
39. a
40. a
41. a
42. a
43. a
44. a
45. a
46. a
47. a
48. a
49. a
50. a
51. a
52. a
53. a
54. a
55. a
56. a
57. a
58. a
59. a
60. a
61. a
62. a
63. a
64. a
65. a
66. a
67. a
68. a
69. a
70. a
71. a
72. a
73. a
74. a
75. a
76. a
77. a
78. a
79. a
80. a
81. a
82. a
83. a
84. a
85. a
86. a
87. a
88. a
89. a
90. a
91. a
92. a
93. a
94. a
95. a
96. a
97. a
98. a
99. a