CORE JAVA PROGRAMMING STATEMENT (1Z0-808)

By www.HadoopExam.com

Note: These instructions should be used with the HadoopExam Apache OOzie: Professional Trainings. Where it is executed and you can do hands on with trainer.

Cloudera CCA175 (Hadoop and Spark Developer Hands-on Certification available with total 90 solved problem scenarios. Click for More Detail)

Cloudera CCPDE575 (Hadoop BigData Data Engineer Professional Hands-on Certification available with total 79 solved problem scenarios. Click for More Detail)

Java Statements: Controlling which section of the code should be executed. You can have condition and then execute it.

If Statement: Conditional Code execution.

```
public class Welcome {
	public static void main(String[] args) {
		int i=10;
		int y=5;

		if(i>y){
			System.out.println("Print  1 : Here I > Y ");
			y=11;
		}

		System.out.println("Print  2 : Value of I = " + i + " Value of Y = " + y);

		if(y>i){
			System.out.println("Print  3 : Here Y > I ");
		}

		//Code will never enter in if condition block as it is always evaluated to false.
		if(i>y){
			System.out.println("Print  4 : Print Here I > Y ");
			y=11;
		}

		if(y>i)
			System.out.println("Print  5 : Here Y > I ");

		System.out.println("Print  6 : End of Program");
	}
}
```

If-else statement: Conditional code branching.

```
public class Welcome {
	public static void main(String[] args) {
		int i=10;
		int y=5;
```

```
            if(i>y){
                  System.out.println("Print  1 : Here I > Y ");
                  y=11;
                  System.out.println("Print  2 : Value of I = " + i + " Value of Y = " + y);
            }
            else if(y>i){ //This else block will never be executed. Even condition is true.
                  System.out.println("Print  3 : Here Y > I ");
            }

            //Code will never enter in if condition block as it is always evaluated to false.
            if(i>y){
                  System.out.println("Print  4 : Print Here I > Y ");
                  y=11;
            }
            else
                  System.out.println("Print  5 : Here Y > I ");

            //Code will never enter in if condition block as it is always evaluated to false.
                        if(i>y){
                              System.out.println("Print  6 : Print Here I > Y ");
                              y=11;
                        }
                        else if(y>i)
                              System.out.println("Print  7 : Here Y > I ");

            /*if(int val=5){ //Will not compile
                  System.out.println("Print  8 ");
            }*/

            System.out.println("Print  9 : End of Program");
      }
}
```

Ternary Operator: **(? :) :** This can work on up to three expression. It is equivalent to if-then-else block.

(Always a Boolean expression) ? (If Boolean expression is true then this section) : (if Boolean expression is false then this section)

```
public class Welcome {
	public static void main(String[] args) {
		//if then else example
		int y = 20;
		int x = 0;
		if (y > 5) {
			x = 3 * y;
		} else {
			x = 4 * y;
		}
		System.out.println("Value of x = " + x + " Value of y = " + y);

		//Ternary operator
		y=20;
		x=0;
		int result = (y>5) ? x= 3*y : 4*y;
		System.out.println("Value of x = " + x + " Value of y = " + y + " And result = " + result);

		//Behaves like short circuit operator
		int val1 = 1;
		int val2 = 1;
		final int val3 = val2<2 ? val1++ : val2++;
		System.out.println(val1+","+val2+ ","+ val3); // Outputs 2,1,1
	}
}
```

switch operator: Complex decision making statement.

- is a complex decision-making structure in which a single value is evaluated and fl ow is redirected to the first matching branch (case statement)
- If no such case statement is found that matches the value, an optional *default* statement will be called.
- If no such default option is available, the entire switch statement will be skipped.
- Data type of switch value and case value should be same.
- In each case, you must have break; statement.
- Even though the default block was before the case block, only the case block was executed.
- default is only branched to if there is no matching case value for the switch statement, regardless of its position within the switch statement.
- the case statement value must also be a literal, enum constant, or final constant variable.

```
public class Welcome {
	public static void main(String[] args) {
		int monthVal=Integer.parseInt(args[0]);
		//Also try with removing break statement.
		switch(monthVal){
			case 1 :
				System.out.println("Jan");
				break;
			case 2 :
				System.out.println("Feb");
				break;
			case 3 :
				System.out.println("Mar");
				break;
			case 4 :
				System.out.println("Apr");
				break;
			default:
				System.out.println("No proper month value");
			case 5 :
				System.out.println("May");
				break;

		}
	}
}
```

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

		String sitename = "HadoopExam";
		String www = "www.";
		String name = "HadoopExam";
		String com = ".com";
		int value = 0;

		switch (name) {
		case "Hadoop":
			break;
		case sitename:
```

```
                    value = 1;
                    break;
            case www:
                    value = 2;
                    break;
            case com:
                    value = 3;
                    break;
            case 5000: // DOES NOT COMPILE
                    value = 4;
                    break;
            case 'H': // DOES NOT COMPILE
                    value = 5;
                    break;
            case Welcome.class: // DOES NOT COMPILE
                    value = 6;
                    break;
            }
      }
}
```

Data types supported by switch statements include the following:
int and Integer
byte and Byte
short and Short
char and Character
int and Integer
String
enum values

while statement: (it is also known as while loop)

- Infinite while loop

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

   while(true){
         System.out.println("I am learning Java from HadoopExam.com");
         }
   }
}
```

- **Controlled while loop**

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

         int i = 0;

         while (i < 10) {
               System.out.println("I am learning Java from HadoopExam.com value of i " + i);
               i++;
         }

         while (i > 10) { //will not pass this condition
               System.out.println("I am learning Java from HadoopExam.com value of i " + i);
               i++;
         }

         int j=5;
         while (i > 10 || j<10) {
               System.out.println("I am learning Java from HadoopExam.com value of i " + i + " And
value of j " + j);
               j++;
         }

         while (i < 11 || j<10) { //Infinite loop
               System.out.println("I am learning Java from HadoopExam.com value of i " + i + " And
value of j " + j);
               j++;
               //if(j>2000) //uncomment and test again
               //break;
         }
```

```
        }
    }
```

do-while loop

- a do-while loop guarantees that the statement or block will be executed at least once.
- It is your choice, which one to use while or do-while.
- It is recommended you use while.

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

                int i = 0;
                do {
                        i++;
                } while (false);
                System.out.println(i);
        }
    }
```

for statement : another controlled looping.

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

                for(int i=0; i<10; i++){
                        int j=0;
                        System.out.println("Value of i = " + i + "Value of j = " + j++);
                }
                //Both i and j are local variable
                //System.out.println("Value of J" + j); // j will not be available here.
                //System.out.println("Value of i" + i); // i will not be available here.
        }
}
```

Infinite for loop

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

		for( ; ; ) {
			System.out.println("Welcome to HadoopExam.com");
			}
	}
}
```

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

		for( int i=0, j=0; i<10 && j<10 ;  ) {
			System.out.println("Value of i " + i + " Value of j " + j);
			i++; j++;
			}
	}
}
```

Below code will not compile: Because you are trying re-initialize the j

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

		int j=0;
		for( int i=0, j=4; i<10 && j<10 ;  ) {
			System.out.println("Value of i " + i + " Value of j " + j);
			i++; j++;
			}
	}
}
```

Incompatible data type in initialization block:

```
public class Welcome {
	public static void main(String[] args) {
		// This code will not compile. As you can not use different data type.
		for(long y = 0, int x = 1; x < 5 && y<10; x++, y++) {
```

```
                    System.out.print(x + " ");
                    }
        }
}
```

for each statement : (It is added only in Java 5.0)

```
for(datatype instance : collection/array)
```

- Collection/array must be iterable : an object whose class implements java.lang.Iterable

```
public class Welcome {
      public static void main(String[] args) {
            int i = 0;
            for (char c : "HadoopExam".toCharArray()) {
                  System.out.println(c);
            }
      }
}
```

- **Array of strings example**

```
public class Welcome {
      public static void main(String[] args) {
            int i = 0;
            for (String siteName : args) {
                  System.out.println(siteName);
            }
      }
}
```

- **Iterating over list**

```
   import java.util.ArrayList;
   import java.util.List;
```

```
   public class Welcome {
      public static void main(String[] args) {
            List<String> list = new ArrayList<String>();
            list.add("HadoopExam.com");
            list.add("QuickTechie.com");
            list.add("Training4Exam.com");

            for (String siteName : list) {
                  System.out.println(siteName);
            }
      }
   }
```

- **For-each loop is just an compile time enhancements. Once code is compiled java internally convert it into normal for loop.**

Nested Loops : Loop inside loop.

```
public class Welcome {
      public static void main(String[] args) {
            for(int i=0; i<5;i++){
                  for(int j=0;j<3;j++){
                        System.out.println("Value of i = " + i + " Value of j = " + j);
                  }
            }
      }
}
```

Formatted output of nested loops.

```
public class Welcome {
      public static void main(String[] args) {
            for(int i=0; i<5;i++){
                  for(int j=0;j<3;j++){
                        System.out.print(i+","+j + "\t");
                  }
                  System.out.println();
            }
      }
}
```

break statement : Break the loop and come out of the loop

- You can use in any loop this break statement while, for, for-each, do-while etc.

```
public class Welcome {
	public static void main(String[] args) {
		int j=0;
		for(int i=0; i<5;i++){
				System.out.println("Value of i " + i);
				j++;
				if(j>3)
				{
					break;
				}
			}
	}
}
```

continue statement :

```
public class Welcome {
	public static void main(String[] args) {
		int j=0;
		for(int i=0; i<5;i++){
				j++;
				if(j==3)
				{
					continue;
				}
				System.out.println("Value of i " + i + " Value of j " + j);
			}
	}
}
```

```
public class Welcome {
	public static void main(String[] args) {
		int j=0;
		for(int i=0; i<5;i++){
				if (i<3){
					continue;
				}
				System.out.println("Inside while loop : value of i = " + i);
			}
	}
}
```

- While the break statement transfers control to the enclosing statement
- the continue statement transfers control to the boolean expression that determines if the loop should continue. it ends the current iteration of the loop.

```
public class Welcome {
	public static void main(String[] args) {
		FIRST_LOOP: for (int count = 1; count <= 4; count++) {
			for (char seq = 'a'; seq <= 'c'; seq++) {
				if (count == 2 || seq == 'b')
					continue FIRST_LOOP;
				System.out.print(" " + count + seq);
			}
		}
	}
}
```

All Products List of www.HadoopExam.com

TRAINING'S

- Hadoop BigData Professional Training (3500INR/$79)
- HBase (NoSQL) Professional Training (3500INR/$79)
- Apache Spark Professional Training (~~3900INR/$89~~ for a week 3500INR/$75)
- Apache OOZie (Hadoop workflow) Professional Training
- Beginner AWS Training Course- **(HETRNAWS101)**
- Core Java 1z0-808 Exam training

MAPR HADOOP AND NOSQL CERTIFICATION

- **MapR Hadoop Developer Certification**
- **MapR HBase NoSQL Certification**
- **MapR Spark Developer Certification (In Progress)**

CLOUDERA HADOOP AND NOSQL CERTIFICATION

- CCA50X : Hadoop Administrator
- CCA-175 Cloudera® (Hadoop and Spark Developer)
- CCP:DE575 : Cloudera® Data Engineer Certification

DATABRICKSA OREILLY SPARK CERTIFICATION

- Apache Spark Developer

AWS: AMAZON WEBSERVICE CERTIFICATION

- AWS Solution Architect : Associate
- AWS Solution Architect: Professional
- AWS Developer : Associate
- AWS Sysops Admin : Associate

MICROSOFT AZURE CERTIFICATION

- Azure 70-532
- Azure 70-533

DATA SCIENCE CERTIFICATION

- EMC E20-007

EMC CERTIFICATIONS

- EMC E20-007

SAS ANALYTICS CERTIFICATION

- SAS Base A00-211
- SAS Advanced A00-212
- SAS Analytics : A00-240
- SAS Administrator : A00-250

ORACLE JAVA CERTIFICATION

- Java 1z0-808
- Java 1z0-809

ORACLE DATABASE CLOUD CERTIFICATION

- 1z0-060 (Oracle 12c)
- 1z0-061 (Oracle 12c)

Subscribe Here for Regular Updates: Like New Training Module launched

Become Author and Trainer: We are looking for Author (Writing Technical Books) and Trainer (Creating Training Material): **No Compromise on Quality.**

Benefit: You will get very good revenue sharing. Please drop us an email to hadoopexam@gmail.com (For the skills, you feel you are master)

We are sure, you are good at least one technology. Don't limit your potential, contact us immediately with your skill. Our expert team will contact you with more detail. You training and Books will reach to all our existing network and with our expert marketing team we will help you to reach as much as technical professional, with our Smart Advertising network. Contact us with sending an email hadoopexam@gmail.com

Opportunity to share your knowledge with all learners who are in need. We are helping 1000's of learners since last 4 years and established ourselves with Quality low cost material.