Types of Control Statements:

Control Statements
1. Definition

- Control statements determine the flow of execution in a program.
- They allow the program to make decisions, repeat actions, and control the order in which statements are executed.

Conditional (Selection) Statements

Direct the flow of execution based on conditions.
Eg:
if Statement
```
if (x > 10) {
   System.out.println("x is greater than 10");
}
```
if-else Statement:
```
if (x > 10) {
   System.out.println("x is greater than 10");
} else {
   System.out.println("x is 10 or less");
}
```

- Direct the flow of execution based on conditions.

Eg:
if Statement
```
if (x > 10) {
   System.out.println("x is greater than 10");
}
```
if-else Statement:
```
if (x > 10) {
   System.out.println("x is greater than 10");
} else {
   System.out.println("x is 10 or less");
}
```

Switch Statement
```
switch (day) {
   case 1: System.out.println("Sunday"); break;
   case 2: System.out.println("Monday"); break;
   // ...
}
```

Jump Statements
Control the flow of execution by jumping to different parts of the program.
Examples:
break: Exits the current loop or switch
```
for (int i = 0; i < 10; i++) {
   if (i == 5) {
      break;  // Exits loop when i is 5
   }
}
```
continue: Skips the current iteration of a loop
```
for (int i = 0; i < 10; i++) {
   if (i == 5) {
      continue;  // Skips the iteration when i is 5
   }

System.out.println(i);
}
```
return: Exits from the current method and optionally returns a value.

Looping (Iteration) Statements
Repeat a block of code as long as a condition is true.

Examples:
for Loop:
```
for (int i = 0; i < 5; i++) {

System.out.println(i);
}
```
while Loop:
```
while (x < 10) {
   x++;
}
```
do-while Loop
```
do {
   x++;
} while (x < 10);
```