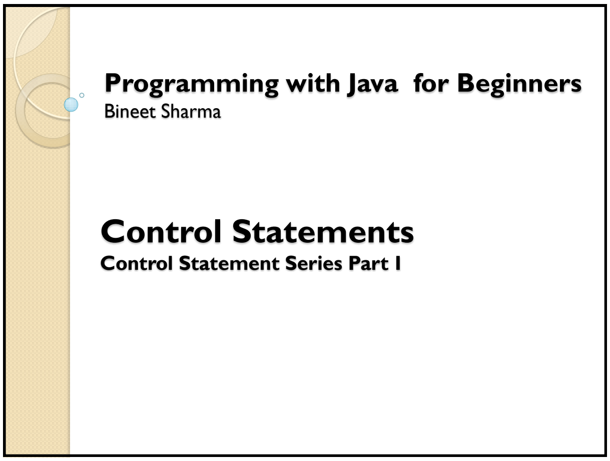
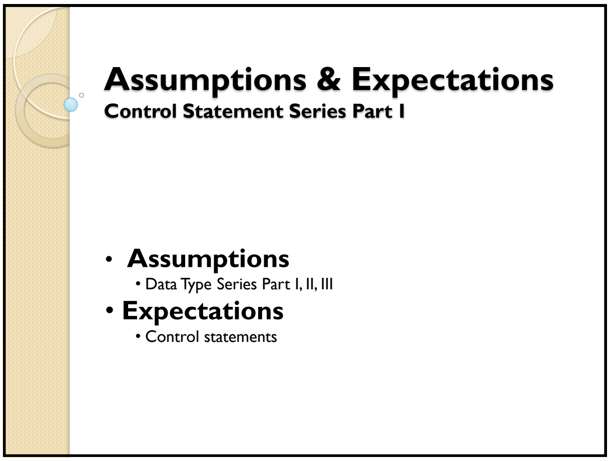


Programming with Java for Beginners

Bineet Sharma

Control Statements

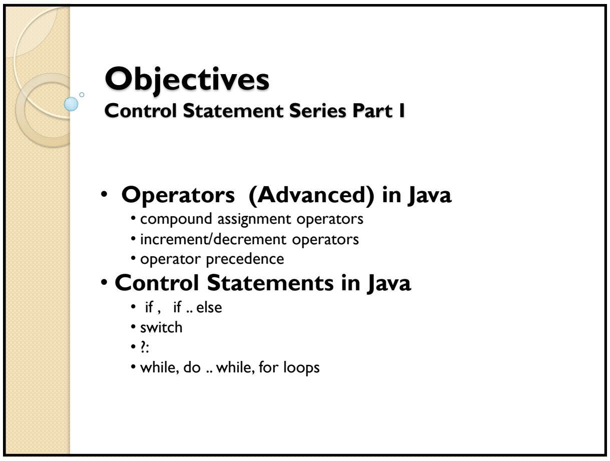
Control Statement Series Part I



Assumptions & Expectations

Control Statement Series Part I

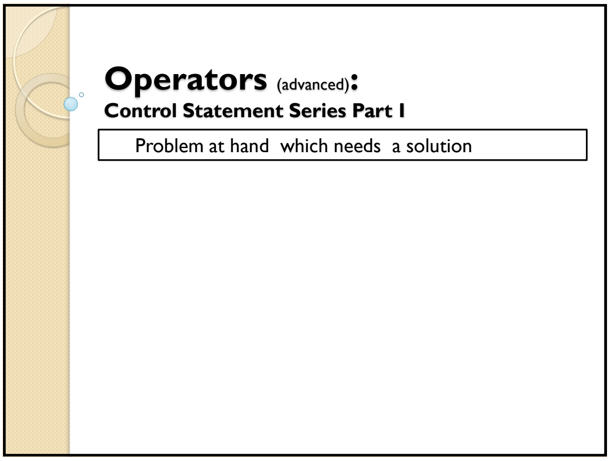
- **Assumptions**
  - Data Type Series Part I, II, III
- **Expectations**
  - Control statements



Objectives

Control Statement Series Part I

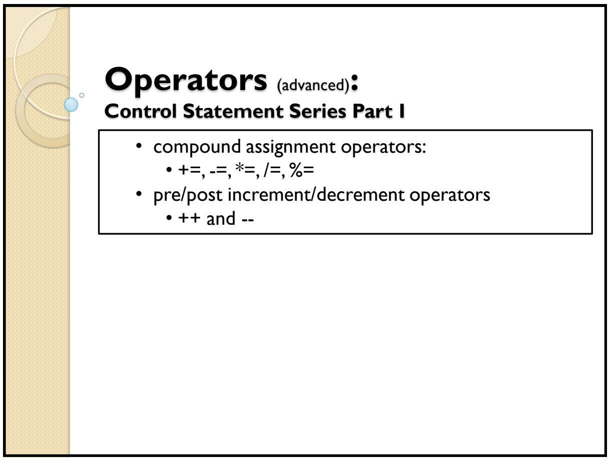
- **Operators (Advanced) in Java**
  - compound assignment operators
  - increment/decrement operators
  - operator precedence
- **Control Statements in Java**
  - if , if .. else
  - switch
  - ?:
  - while, do .. while, for loops



Operators (advanced):

Control Statement Series Part I

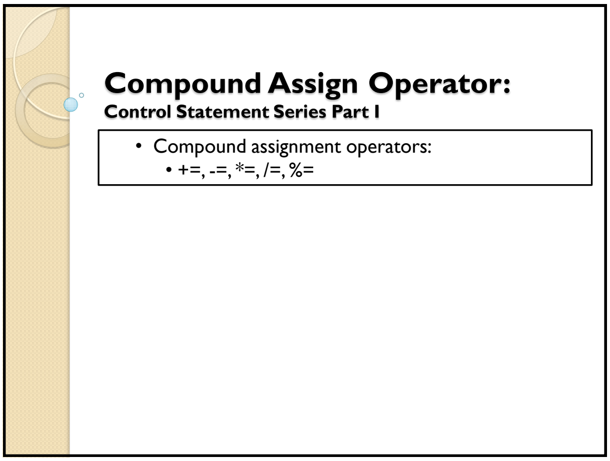
Problem at hand which needs a solution



Operators (advanced):

Control Statement Series Part I

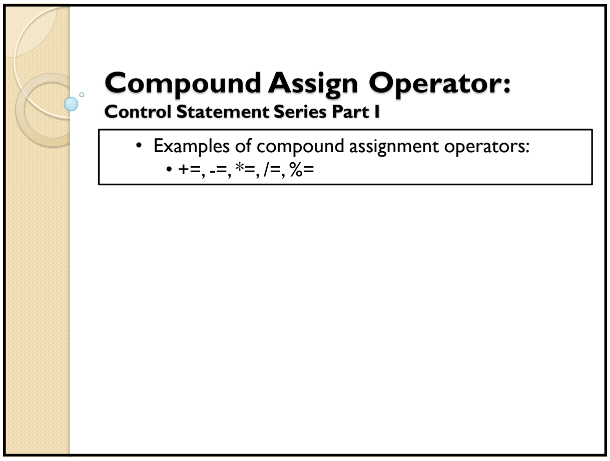
- compound assignment operators:
  - +=, -=, *=, /=, %=
- pre/post increment/decrement operators
  - ++ and --



Compound Assign Operator:

Control Statement Series Part I

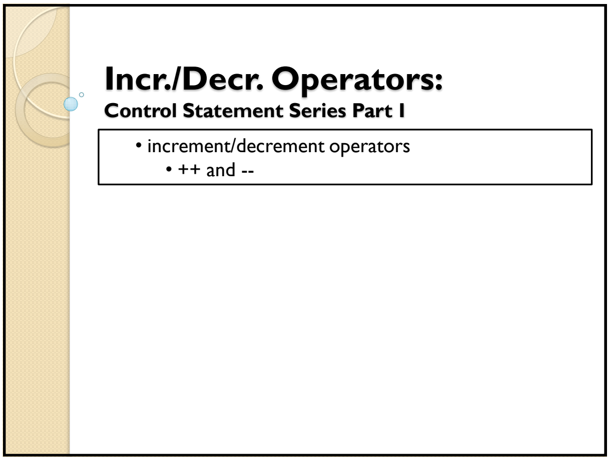
- Compound assignment operators:
  - +=, -=, *=, /=, %=



Compound Assign Operator:

Control Statement Series Part I

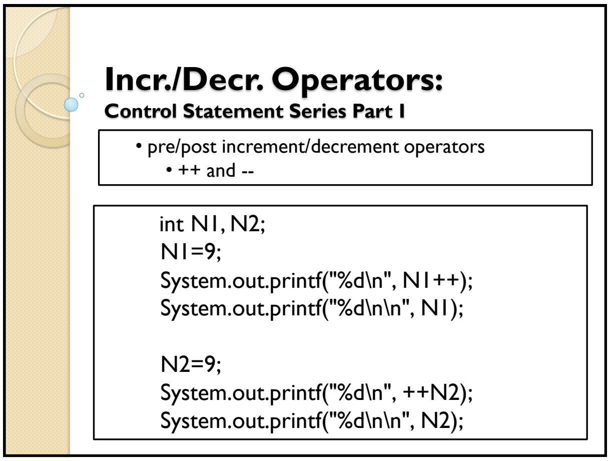
- Examples of compound assignment operators:
  - +=, -=, *=, /=, %=



Incr./Decr. Operators:

Control Statement Series Part I

- increment/decrement operators
  - ++ and --

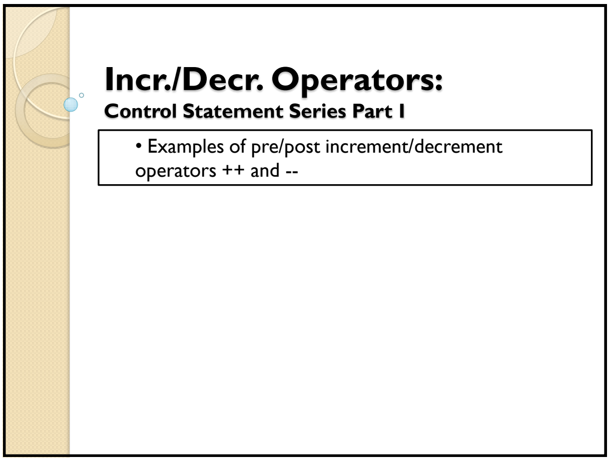


Incr./Decr. Operators:

Control Statement Series Part I

- pre/post increment/decrement operators
  - ++ and --

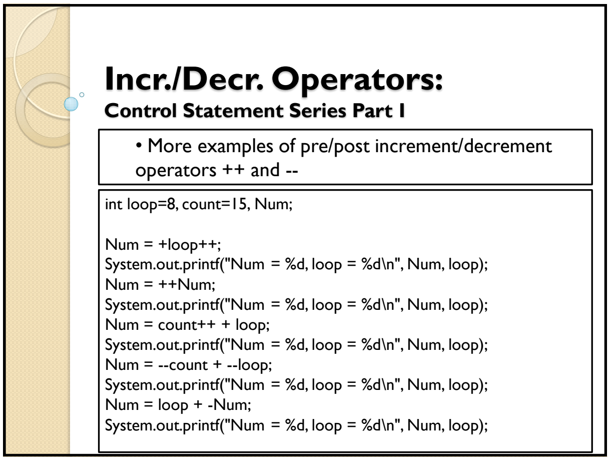
```
int N1, N2;
N1=9;
System.out.printf("%d\n", N1++);
System.out.printf("%d\n\n", N1);

N2=9;
System.out.printf("%d\n", ++N2);
System.out.printf("%d\n\n", N2);
```



Incr./Decr. Operators:

Control Statement Series Part I

- Examples of pre/post increment/decrement operators ++ and --

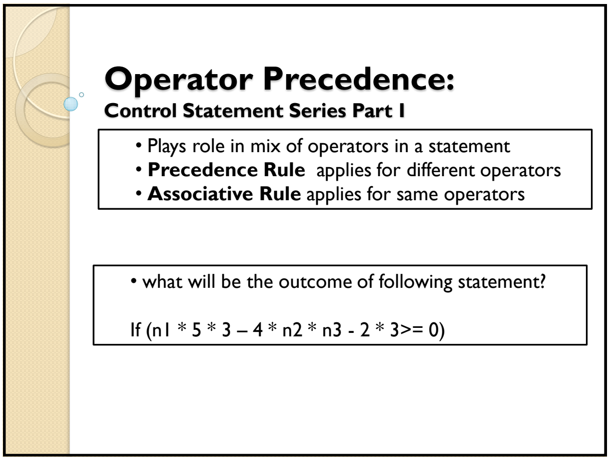


Incr./Decr. Operators:

Control Statement Series Part I

- More examples of pre/post increment/decrement operators ++ and --

```
int loop=8, count=15, Num;

Num = +loop++;
System.out.printf("Num  = %d, loop = %d\n", Num, loop);
Num = ++Num;
System.out.printf("Num  = %d, loop = %d\n", Num, loop);
Num = count++ + loop;
System.out.printf("Num  = %d, loop = %d\n", Num, loop);
Num = --count + --loop;
System.out.printf("Num  = %d, loop = %d\n", Num, loop);
Num = loop + -Num;
System.out.printf("Num  = %d, loop = %d\n", Num, loop);
```



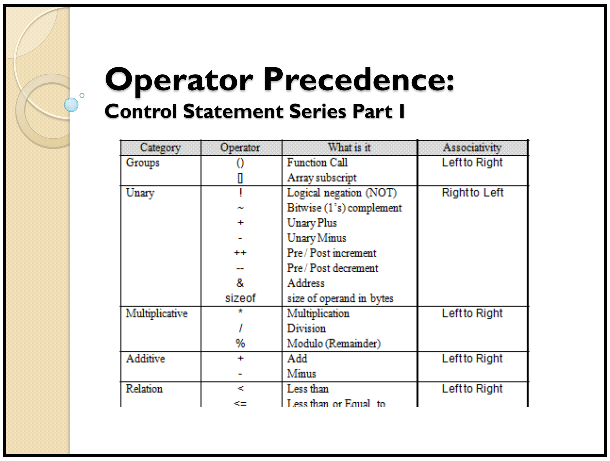
Operator Precedence:

Control Statement Series Part I

- Plays role in mix of operators in a statement
- **Precedence Rule** applies for different operators
- **Associative Rule** applies for same operators

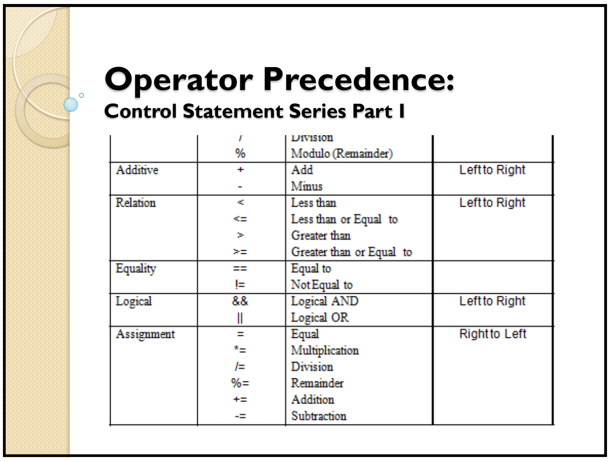
- what will be the outcome of following statement?

If (n1 * 5 * 3 – 4 * n2 * n3 - 2 * 3>= 0)



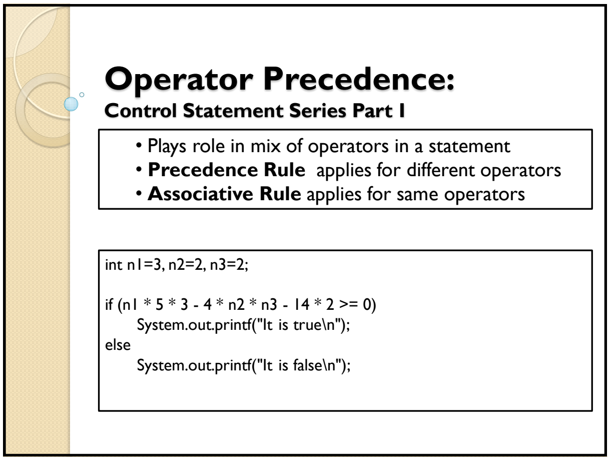
Operator Precedence:
Control Statement Series Part I

Category	Operator	What is it	Associativity
Groups	() []	Function Call Array subscript	Left to Right
Unary	! ~ + - ++ -- & sizeof	Logical negation (NOT) Bitwise (1's) complement Unary Plus Unary Minus Pre / Post increment Pre / Post decrement Address size of operand in bytes	Right to Left
Multiplicative	* / %	Multiplication Division Modulo (Remainder)	Left to Right
Additive	+ -	Add Minus	Left to Right
Relation	< <=	Less than Less than or Equal to	Left to Right



Operator Precedence:
Control Statement Series Part I

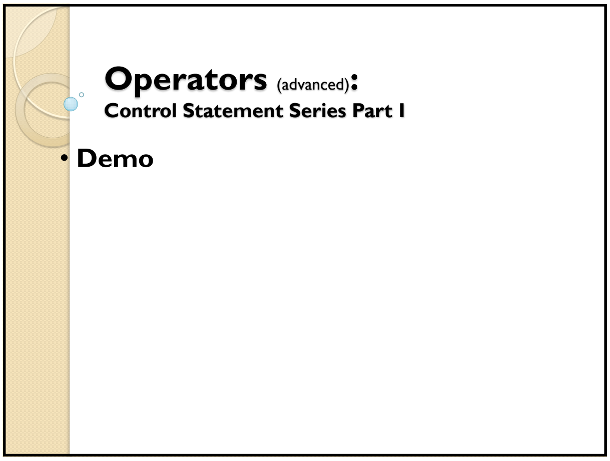
	/ %	Division Modulo (Remainder)	
Additive	+ -	Add Minus	Left to Right
Relation	< <= > >=	Less than Less than or Equal to Greater than Greater than or Equal to	Left to Right
Equality	== !=	Equal to Not Equal to	
Logical	&& ||	Logical AND Logical OR	Left to Right
Assignment	= *= /= %= += -=	Equal Multiplication Division Remainder Addition Subtraction	Right to Left



Operator Precedence:
Control Statement Series Part I

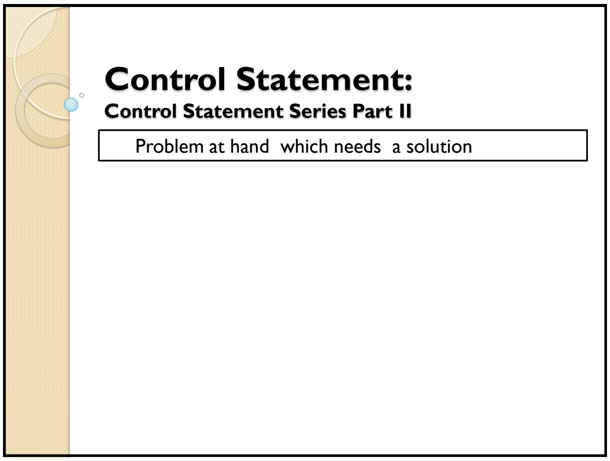
- Plays role in mix of operators in a statement
- **Precedence Rule** applies for different operators
- **Associative Rule** applies for same operators

```
int n1=3, n2=2, n3=2;

if (n1 * 5 * 3 - 4 * n2 * n3 - 14 * 2 >= 0)
      System.out.printf("It is true\n");
else
      System.out.printf("It is false\n");
```



Operators (advanced):
Control Statement Series Part I

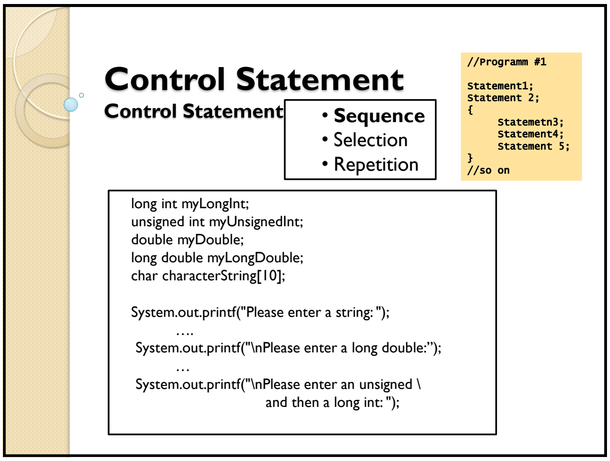
- **Demo**



Control Statement:

Control Statement Series Part II

Problem at hand which needs a solution



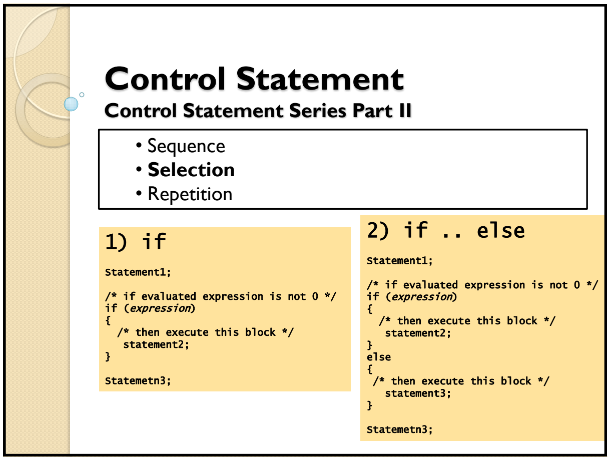
Control Statement

Control Statement

- **Sequence**
- Selection
- Repetition

```
//Programm #1

Statement1;
Statement 2;
{
      Statemetn3;
      Statement4;
      Statement 5;
}
//so on
```

```
long int myLongInt;
unsigned int myUnsignedInt;
double myDouble;
long double myLongDouble;
char characterString[10];

System.out.printf("Please enter a string: ");
          ….
 System.out.printf("\nPlease enter a long double:");
          …
 System.out.printf("\nPlease enter an unsigned \
                              and then a long int: ");
```



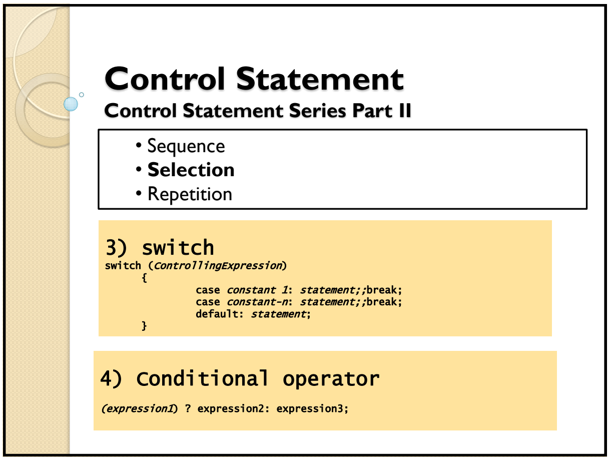
Control Statement

Control Statement Series Part II

- Sequence
- **Selection**
- Repetition

```
1) if

Statement1;

/* if evaluated expression is not 0 */
if (expression)
{
  /* then execute this block */
   statement2;
}

Statemetn3;
```

```
2) if .. else

Statement1;

/* if evaluated expression is not 0 */
if (expression)
{
  /* then execute this block */
   statement2;
}
else
{
 /* then execute this block */
   statement3;
}

Statemetn3;
```



Control Statement

Control Statement Series Part II

- Sequence
- **Selection**
- Repetition

```
3) switch
switch (ControllingExpression)
      {
               case constant 1: statement;;break;
               case constant-n: statement;;break;
               default: statement;
      }
```

```
4) Conditional operator

(expression1) ? expression2: expression3;
```

Control Statement

Control Statement Series Part II

- Sequence
- **Selection**
- Repetition

'if' construct Example:

```
….
if (age <= 10)
          System.out.printf ("you are a kid");
….
if (age >= 18)
          System.out.printf ("you are adult, you can vote");
…
```

Control Statement

Control Statement Series Part II

- Sequence
- **Selection**
- Repetition

'if' construct Example: *Would this work?*

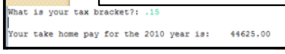
```
….
if (age <= 10)
          System.out.printf ("you are a kid");
….
if (age >= 18)
          System.out.printf ("you are adult\n");
          System.out.printf ("you can vote");
…
```

Control Statement

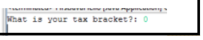
Control Statement Series Part II

Example of **'if'** statement

```
double  yourSalary=50000, yourBonus=2500;
double yourTakeHomePay;
double yourTaxBracket;
Scanner readInput = new Scanner(System.in);
System.out.printf("What is your tax bracket?: ");
yourTaxBracket= readInput.nextDouble();
if (yourTaxBracket > 0) {
     yourTakeHomePay = (yourSalary + yourBonus)* (1-yourTaxBracket);
     System.out.printf("\nYour take home pay for the 2010 year
                is:  %10.2f\n\n",  yourTakeHomePay);
}
```



```
What is your tax bracket?: .15

Your take home pay for the 2010 year is:    44625.00
```



```
What is your tax bracket?: 0
```

Control Statement

Control Statement Series Part II

- Sequence
- **Selection**
- Repetition

'if .. else' construct

```
Statement1;

// if evaluated expression is not 0
if (expression)
{  // then execute this block
   statement2;
}
else
{   // otherwise execute this block
   statement3;
}

Statemetnt4;
```

Control Statement

Control Statement Series Part II

Example of '**if .. else** ' statement

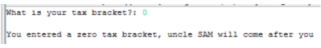
```
double  yourSalary=50000, yourBonus=2500;
….
…
yourTaxBracket= readInput.nextDouble();
if (yourTaxBracket > 0) {
     yourTakeHomePay = (yourSalary + yourBonus)* (1-
     yourTaxBracket);
     System.out.printf("\nYour take home pay for the 2010 year
                    is:  %10.2f\n\n",  yourTakeHomePay);
}
else {
     System.out.printf("\nYou entered a zero tax bracket, uncle SAM
     will come after you\n");
}
```

Control Statement

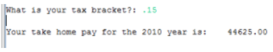
Control Statement Series Part II

Example of '**if .. else** ' statement

```
double  yourSalary=50000, yourBonus=2500;
….
…
yourTaxBracket= readInput.nextDouble();
if (yourTaxBracket > 0) {
     ….
}
else {
     …
}
```



```
What is your tax bracket?: 0

You entered a zero tax bracket, uncle SAM will come after you
```



```
What is your tax bracket?: .15

Your take home pay for the 2010 year is:    44625.00
```

Control Statement:

Control Statement Series Part II

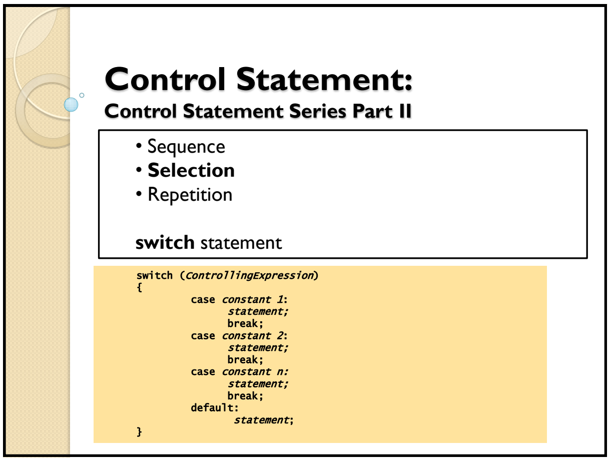
Nested **if**

Control Statement:

Control Statement Series Part II

Nested **if..else**

```
int studentScore;
Scanner readInput = new Scanner(System.in);

System.out.printf("Please enter student's score: ");
studentScore= readInput.nextInt();

if (studentScore >= 90)
          printf("A\n");
else if (studentScore >= 80)
          printf("B\n");
else if (studentScore >= 70)
          printf("C\n");
else if (studentScore >= 60)
          printf("D\n");
else
          printf("F\n");
```



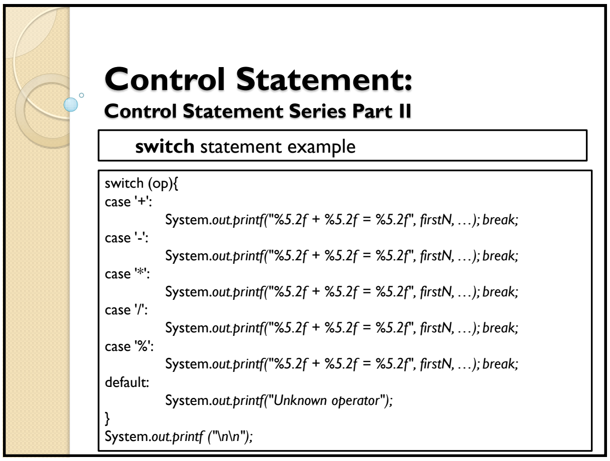
Control Statement:

Control Statement Series Part II

- Sequence
- **Selection**
- Repetition

switch statement

```
switch (ControllingExpression)
{
          case constant 1:
                statement;
                break;
          case constant 2:
                statement;
                break;
          case constant n:
                statement;
                break;
          default:
                 statement;
}
```

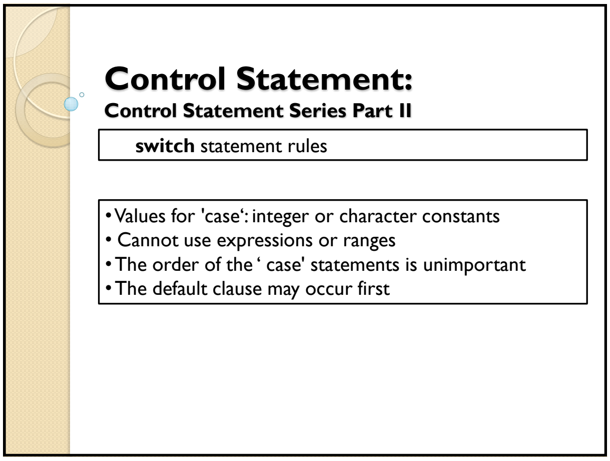


Control Statement:

Control Statement Series Part II

switch statement example

```
switch (op){
case '+':
            System.out.printf("%5.2f + %5.2f = %5.2f", firstN, …); break;
case '-':
            System.out.printf("%5.2f + %5.2f = %5.2f", firstN, …); break;
case '*':
            System.out.printf("%5.2f + %5.2f = %5.2f", firstN, …); break;
case '/':
            System.out.printf("%5.2f + %5.2f = %5.2f", firstN, …); break;
case '%':
            System.out.printf("%5.2f + %5.2f = %5.2f", firstN, …); break;
default:
            System.out.printf("Unknown operator");
}
System.out.printf ("\n\n");
```

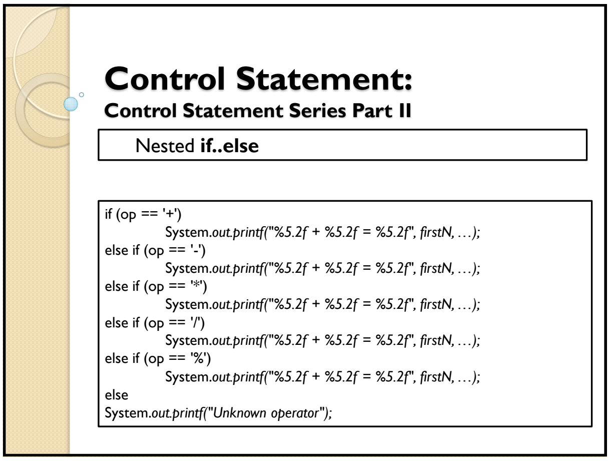


Control Statement:

Control Statement Series Part II

switch statement rules

- Values for 'case': integer or character constants
- Cannot use expressions or ranges
- The order of the ' case' statements is unimportant
- The default clause may occur first

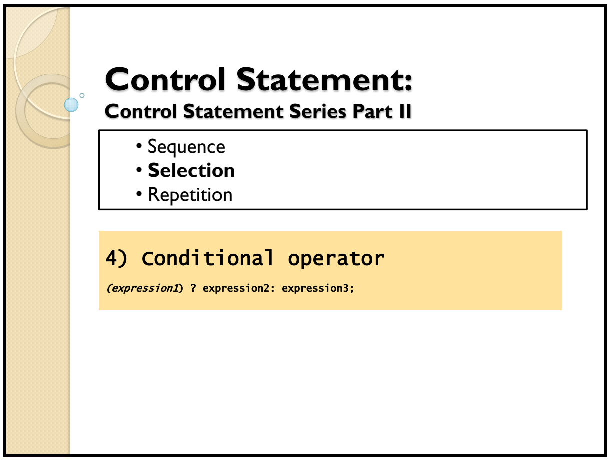


Control Statement:

Control Statement Series Part II

Nested **if..else**

```
if (op == '+')
            System.out.printf("%5.2f + %5.2f = %5.2f", firstN, …);
else if (op == '-')
            System.out.printf("%5.2f + %5.2f = %5.2f", firstN, …);
else if (op == '*')
            System.out.printf("%5.2f + %5.2f = %5.2f", firstN, …);
else if (op == '/')
            System.out.printf("%5.2f + %5.2f = %5.2f", firstN, …);
else if (op == '%')
            System.out.printf("%5.2f + %5.2f = %5.2f", firstN, …);
else
System.out.printf("Unknown operator");
```

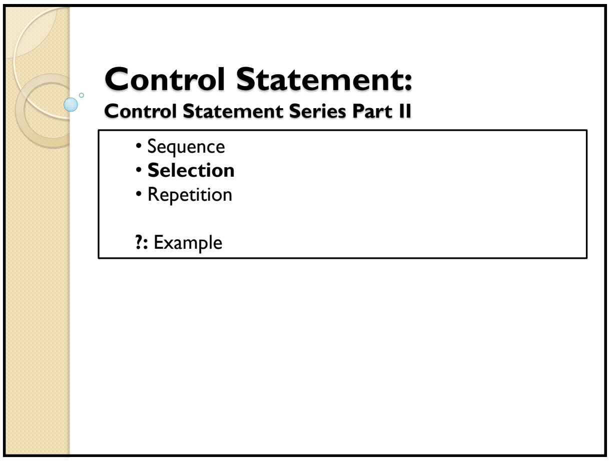


Control Statement:

Control Statement Series Part II

- Sequence
- **Selection**
- Repetition

```
4) Conditional operator
(expression1) ? expression2: expression3;
```

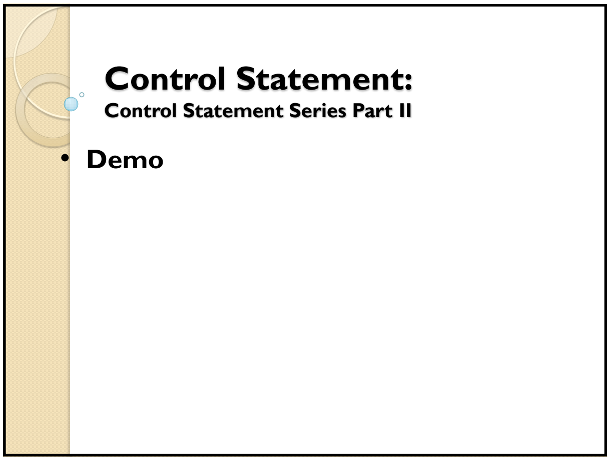


Control Statement:

Control Statement Series Part II

- Sequence
- **Selection**
- Repetition

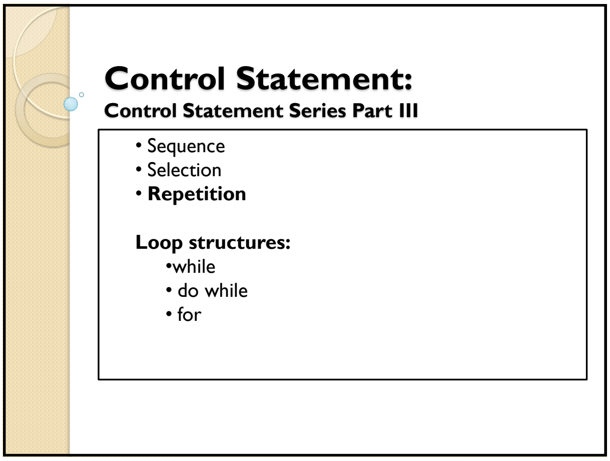
?: Example



Control Statement:

Control Statement Series Part II

- **Demo**



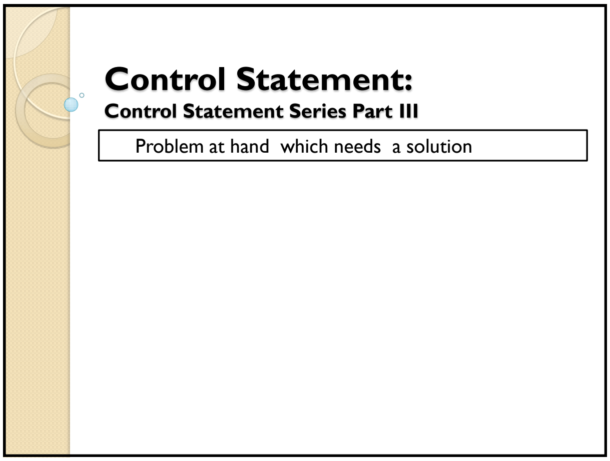
Control Statement:

Control Statement Series Part III

- Sequence
- Selection
- **Repetition**

Loop structures:

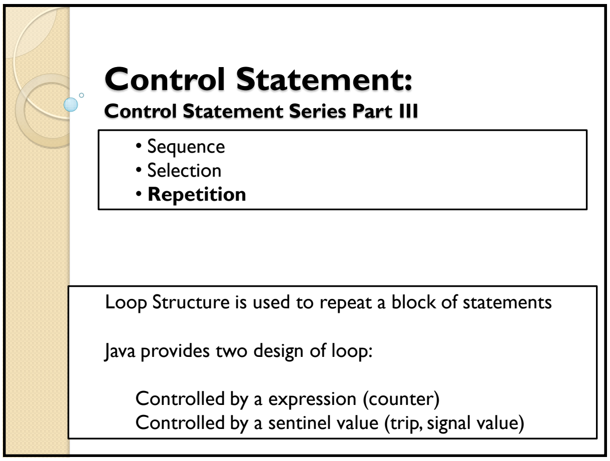
- while
- do while
- for



Control Statement:

Control Statement Series Part III

Problem at hand which needs a solution



Control Statement:

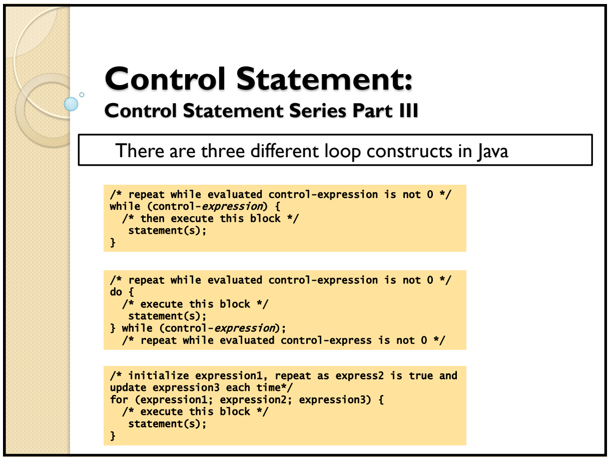
Control Statement Series Part III

- Sequence
- Selection
- **Repetition**

Loop Structure is used to repeat a block of statements

Java provides two design of loop:

Controlled by a expression (counter)
Controlled by a sentinel value (trip, signal value)



Control Statement:

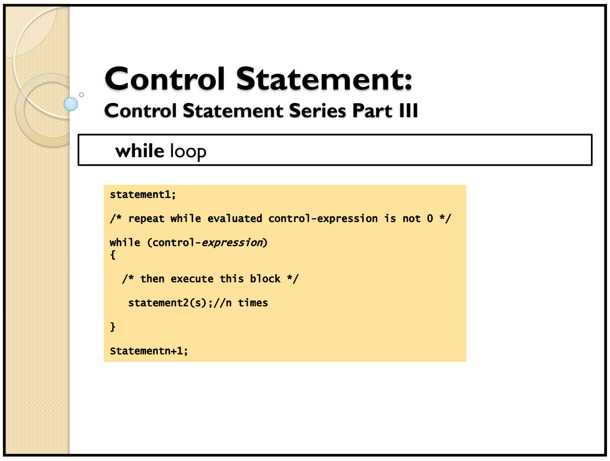
Control Statement Series Part III

There are three different loop constructs in Java

```
/* repeat while evaluated control-expression is not 0 */
while (control-expression) {
  /* then execute this block */
   statement(s);
}
```

```
/* repeat while evaluated control-expression is not 0 */
do {
  /* execute this block */
   statement(s);
} while (control-expression);
  /* repeat while evaluated control-express is not 0 */
```

```
/* initialize expression1, repeat as express2 is true and
update expression3 each time*/
for (expression1; expression2; expression3) {
  /* execute this block */
   statement(s);
}
```



Control Statement:

Control Statement Series Part III

while loop

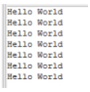
```
statement1;

/* repeat while evaluated control-expression is not 0 */

while (control-expression)
{

  /* then execute this block */

   statement2(s);//n times

}

Statementn+1;
```

Control Statement:

Control Statement Series Part III

while loop example

```
{
      int loop=0;
      while (loop ++ < 10)
            System.out.println("Hello World");
}
```



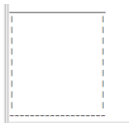
```
Hello World
Hello World
Hello World
Hello World
Hello World
Hello World
Hello World
```

Control Statement:

Control Statement Series Part III

while loop example:

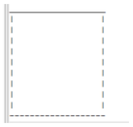
Exercise: Let us write a program to create a box of 20 columns width and 10 columns height using while loop. Use '_' and '|' for line.



Control Statement:

Control Statement Series Part III

while loop example:



Control Statement:

Control Statement Series Part III

do .. while loop

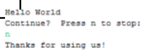
```
Statement1(s);

/* repeat while evaluated control-expression is not 0 */

do
{
  /* execute this block */

   statement2(s); //statement is executed n times

} while (control-expression);
  /* repeat while evaluated control-express is not 0 */

Statement3(s);
```

Control Statement:

Control Statement Series Part III

do .. while loop example
Sentinel controlled

```
int loopCount=0;
char charResponse=' ';
Scanner readInput = new Scanner(System.in);
do
{
      System.out.println("Hello World");
      System.out.println("Continue?  Press n to stop:");
      charResponse = readInput.next().charAt(0);
} while (charResponse != 'n');
System.out.println("Thanks for using us!");
```

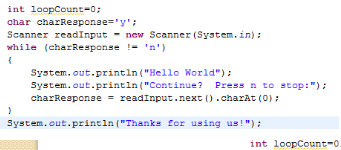


```
Hello World
Continue?  Press n to stop:
n
Thanks for using us!
```

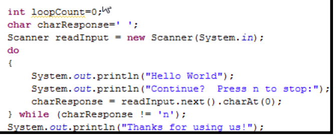
Control Statement:

Control Statement Series Part III

do .. while loop example
-while can be changed to **do while**



```
int loopCount=0;
char charResponse='y';
Scanner readInput = new Scanner(System.in);
while (charResponse != 'n')
{
    System.out.println("Hello World");
    System.out.println("Continue?  Press n to stop:");
    charResponse = readInput.next().charAt(0);
}
System.out.println("Thanks for using us!");
```



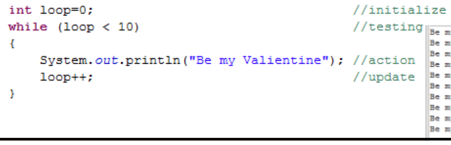
```
int loopCount=0;
char charResponse=' ';
Scanner readInput = new Scanner(System.in);
do
{
    System.out.println("Hello World");
    System.out.println("Continue?  Press n to stop:");
    charResponse = readInput.next().charAt(0);
} while (charResponse != 'n');
System.out.println("Thanks for using us!");
```

Control Statement:

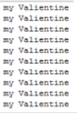
Control Statement Series Part III

Loops : Indefinite and Counting

Three actions needed for counting loops:
- Counter is initialized
- Counter is compared
- Counter is incremented



```
int loop=0;                                    //initialize
while (loop < 10)                              //testing
{
    System.out.println("Be my Valientine");   //action
    loop++;                                    //update
}
```



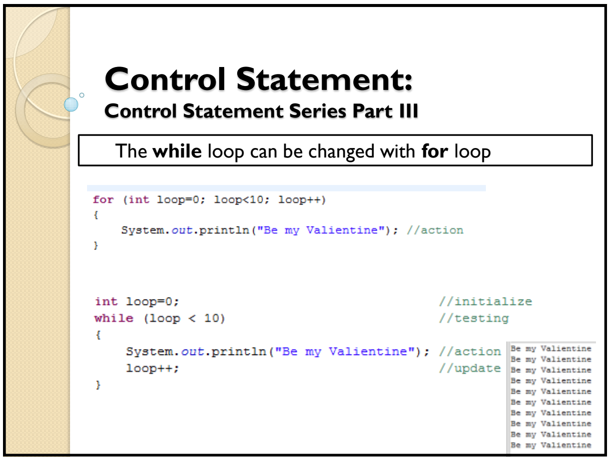
```
Be my Valientine
Be my Valientine
Be my Valientine
Be my Valientine
Be my Valientine
Be my Valientine
Be my Valientine
Be my Valientine
Be my Valientine
Be my Valientine
```

Control Statement:

Control Statement Series Part III

for loop puts those three requirements in one place

```
for (InitializeExpression;
     ControlExpression;
     UpdateExpression)
{
          BlockStatement(s);
}
```



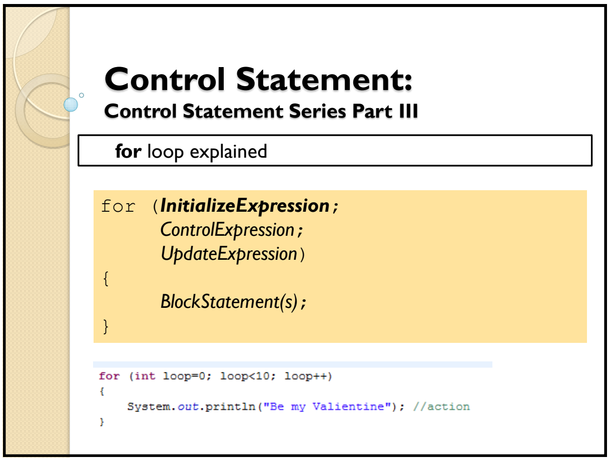
Control Statement:

Control Statement Series Part III

The **while** loop can be changed with **for** loop

```
for (int loop=0; loop<10; loop++)
{
    System.out.println("Be my Valientine"); //action
}


int loop=0;                                   //initialize
while (loop < 10)                             //testing
{
    System.out.println("Be my Valientine");  //action
    loop++;                                   //update
}
```

```
Be my Valientine
Be my Valientine
Be my Valientine
Be my Valientine
Be my Valientine
Be my Valientine
Be my Valientine
Be my Valientine
Be my Valientine
Be my Valientine
```



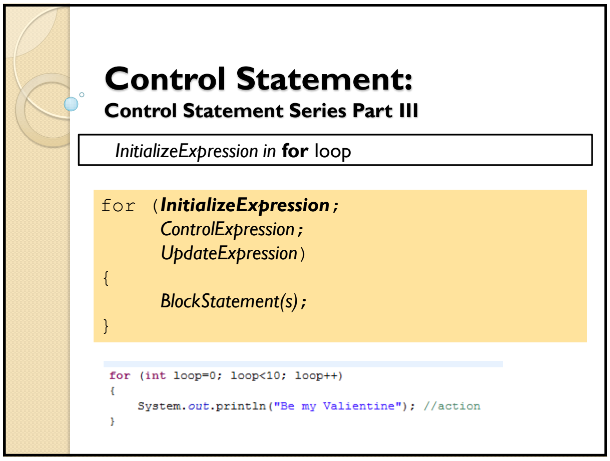
Control Statement:

Control Statement Series Part III

for loop explained

```
for  (InitializeExpression;
      ControlExpression;
      UpdateExpression)
{
      BlockStatement(s);
}
```

```
for (int loop=0; loop<10; loop++)
{
    System.out.println("Be my Valientine"); //action
}
```



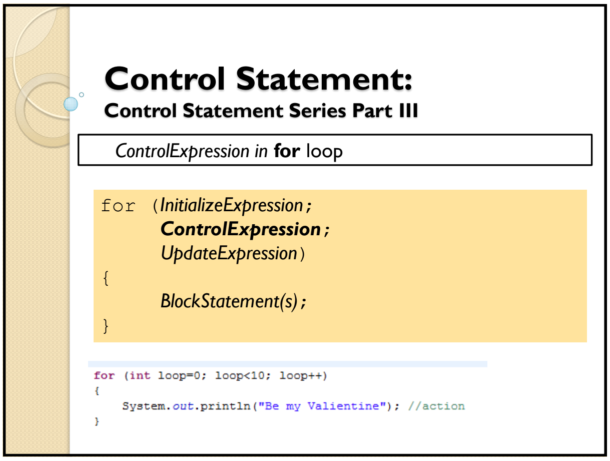
Control Statement:

Control Statement Series Part III

InitializeExpression in **for** loop

```
for  (InitializeExpression;
      ControlExpression;
      UpdateExpression)
{
      BlockStatement(s);
}
```

```
for (int loop=0; loop<10; loop++)
{
    System.out.println("Be my Valientine"); //action
}
```



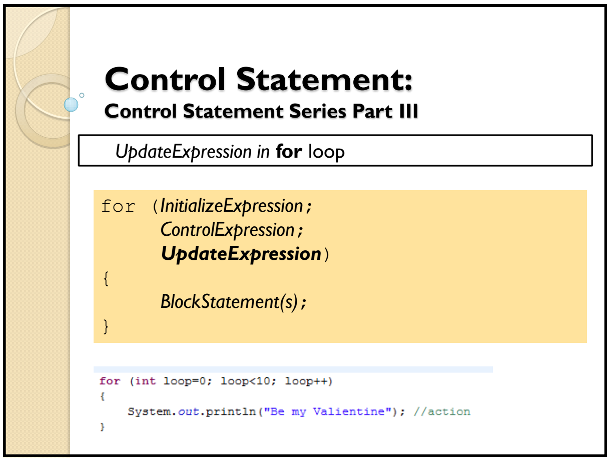
Control Statement:

Control Statement Series Part III

ControlExpression in **for** loop

```
for  (InitializeExpression;
      ControlExpression;
      UpdateExpression)
{
      BlockStatement(s);
}
```

```
for (int loop=0; loop<10; loop++)
{
    System.out.println("Be my Valientine"); //action
}
```



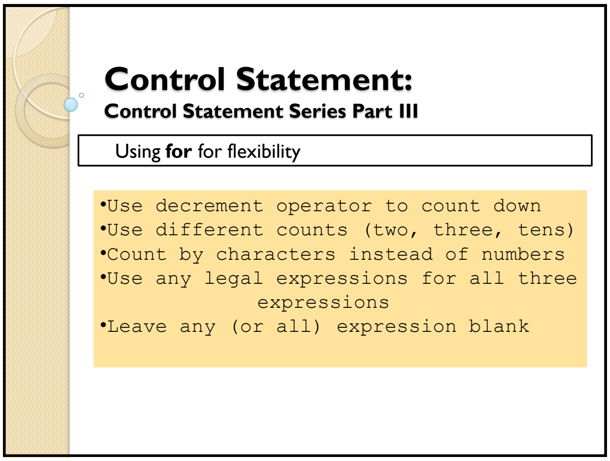
Control Statement:

Control Statement Series Part III

UpdateExpression in **for** loop

```
for  (InitializeExpression ;
        ControlExpression ;
        UpdateExpression)
{
        BlockStatement(s) ;
}
```

```
for (int loop=0; loop<10; loop++)
{
    System.out.println("Be my Valientine"); //action
}
```

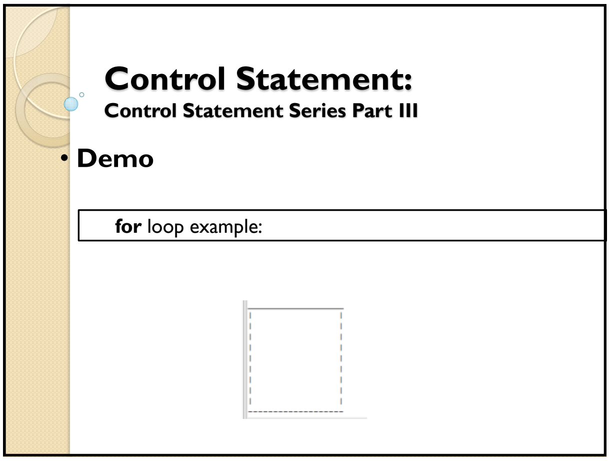


Control Statement:

Control Statement Series Part III

Using **for** for flexibility

- Use decrement operator to count down
- Use different counts (two, three, tens)
- Count by characters instead of numbers
- Use any legal expressions for all three expressions
- Leave any (or all) expression blank

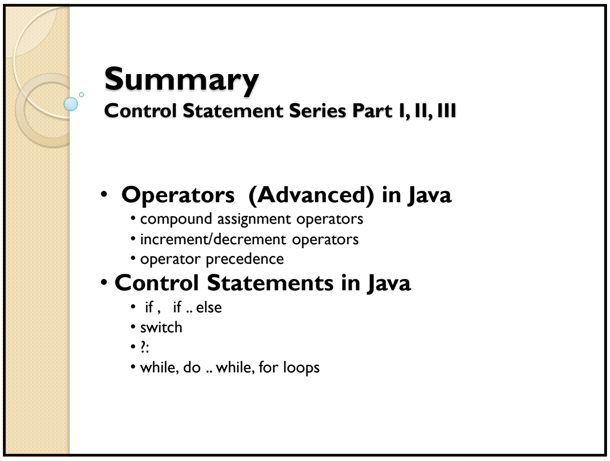


Control Statement:

Control Statement Series Part III

- **Demo**

for loop example:



Summary

Control Statement Series Part I, II, III

- **Operators (Advanced) in Java**
  - compound assignment operators
  - increment/decrement operators
  - operator precedence
- **Control Statements in Java**
  - if , if .. else
  - switch
  - ?:
  - while, do .. while, for loops