Java Programming

Arthur Hoskey, Ph.D.
Farmingdale State College
Computer Systems Department

- Go over important dates.

Important Dates

Today’s Lecture

- If
- If…else
- Nested if
- Dangling else
- Conditional operator
- While loops
- Compound assignment operators
- Increment and decrement operators
- Pre and post increment and decrement

Operators

- Commonly used Java operators (similar to other programming languages).

- + addition
- - subtraction
- * multiplication
- / division
- % mod (this is remainder) – 34 % 10 is 4.

Order of Operations

- Java order of operations is similar to other programming languages.

- () has high precedence
- * /
- + - Low precedednce

- If the condition is true then do something

```
public static void main(String [] args)
{
        int grade = 100;
        if (grade > 89)
        {                                   // Starts if block
                System.out.printf("A");
        }                                   // Ends if block
}
```

If Statements

- Executes ALL statements in the block if the condition is true. What is the output?

```
public static void main(String [] args)
{
        int grade = 100;
        if (grade > 89)
        {
                System.out.printf("A");
                System.out.printf("Super");
                System.out.printf("Spectacular");
        }
}
```

If Statements

If Statements

- You are not required to use a block for the true condition.
- If you do NOT use a block only the first statement after the if is part of the true condition

```
public static void main(String [] args)
{
        int grade = 100;
        if (grade > 89)
                System.out.printf("A");
}
```

What is the output?

```
public static void main(String [] args)
{
        int grade = 95;
        if (grade > 89)
                System.out.printf("A");

        System.out.printf("Super");
        System.out.printf("Spectacular");
}
```

If Statements

What is the output? Is it the same as the previous slide?

```
public static void main(String [] args)
{
        int grade = 80;
        if (grade > 89)
                System.out.printf("A");
                System.out.printf("Super");
                System.out.printf("Spectacular");
}
```

If Statements

If Statements

- My intent was to execute the print statements when grade > 89 but that is not what happens.
- When not using a block only the first statement is part of the if.

```
if (grade > 89)
        System.out.printf("A");
        System.out.printf("Super");
        System.out.printf("Spectacular");
```

NOT part of the if statement.

- Indenting doesn’t make something part of a block, brackets do.

- If the condition is true then do something otherwise do something else.

```
public static void main(String [] args)
{
        if (grade > 89)
        {
                System.out.printf("A");
        }
        else
        {
                System.out.printf("You didn’t get an A");
        }
}
```

If..Else Statements

If..Else Statements

- NOT required to use a block { }.

```
public static void main(String [] args)
{
        int grade = 95;
        if (grade > 89)
                System.out.printf("A");
        else
                System.out.printf("You didn’t get an A");
}
```

- NOT required to use a block { }.
- What is the output?

```
public static void main(String [] args)
{
        int grade = 95;
        if (grade > 89)
                System.out.printf("A");
        else
                System.out.printf("You didn’t get an A");
                System.out.printf("Bad");
                System.out.printf("Horrible");
}
```

If..Else Statements

- Prints "Bad" and "Horrible" which is NOT what I wanted when I indented.
- "Bad" and "Horrible" print statements are NOT part of the else.

```
public static void main(String [] args)
{
        int grade = 95;
        if (grade > 89)
                System.out.printf("A");
        else
                System.out.printf("You didn’t get an A");
                System.out.printf("Bad");
                System.out.printf("Horrible");
}
```

If..Else Statements

switch Statement

- What is the output?

```
public static void main(String [] args)
{
	int num=5;
	switch(num)
	{
		case 1:
			System.out.println("Num is 1");
			break;
		case 2:
			System.out.println("Num is 2");
			break;
		default:
			System.out.println("Num is not 1 or 2");
			break;
	}
}
```

Java switch is the same as C++

- What is the output?

```
public static void main(String [] args)
{
        int num=5;
        switch(num)
        {
                case 1:
                        System.out.println("Num is 1");
                        break;
                case 2:
                        System.out.println("Num is 2");
                        break;
                default:
                        System.out.println("Num is not 1 or 2");
                        break;
        }
}
```

Answer

Num is not 1 or 2

switch Statement

Evaluating Relational Operators

Java relational operators are the same as C++

- num1 == num2

True if num1 and num2 are the *same*

- num1 != num2

True if num1 and num2 are *different*

- num1 > num2

True if num1 is greater than num2

- num1 < num2

True if num1 is less than num2

- num1 >= num2
True if num1 is greater than or equal to num2

- num1 <= num2
True if num1 is less than or equal to num2

Evaluating Relational Operators

Nested If Statements

- One if statement INSIDE of another

```
public static void main(String [] args)
{
        int grade = 80;
        if (grade >= 90)
                System.out.printf("A");
        else
                if (grade >= 80)
                        System.out.printf("B");
                else
                        System.out.printf("Less than B");
}
```

```
public static void main(String [] args)
{           // Larger Example
            int grade = 80;
            if (grade >= 90)
                        System.out.printf("A");
            else
                        if (grade >= 80)
                                    System.out.printf("B");
                        else
                                    if (grade >= 70)
                                                System.out.printf("C");
                                    else
                                                if (grade >= 65)
                                                            System.out.printf("D");
                                                else
                                                            System.out.printf("F");
}
```

Nested If Statements

- What is the output?

```
public static void main(String [] args)
{
   int x = 1, y=10;
   if (x > 5)
          if (y > 5)
                    System.out.println("x and y are > 5");
   else
          System.out.println("x is <= 5");
}
```

Dangling Else

- **Doesn’t print anything!!!**
- Seems like the else is associated with "if (x<5).." because of the way it is indented but it is not.

```
public static void main(String [] args)
{
   int x = 1, y=10;
   if (x > 5)
          if (y > 5)
                    System.out.println("x and y are > 5");
   else
          System.out.println("x is <= 5");
}
```

Dangling Else

Dangling Else

- Else is associated with the if (y > 5)
- **Rule:** Java compiler associates an else with the ***immediately*** preceding if ***unless*** told to do otherwise by the placement of braces { }.

```
public static void main(String [] args)
{
   int x = 1, y=10;
   if (x > 5)
         if (y > 5)
                 System.out.println("x and y are > 5");
   else
         System.out.println("x is <= 5");
}
```

Dangling Else

- How can I force the else to be associated with the "if (x > 5)"?
- Add braces!!!

```
public static void main(String [] args)
{
    int x = 1, y=10;
    if (x > 5)
    {
            if (y > 5)
                    System.out.println("x and y are > 5");
    }
    else
            System.out.println("x is <= 5");
}
```

- Can be used in place of an if…else statement
- Only ternary operator in Java (takes 3 arguments)
- Forms a conditional expression
- Takes the following form:

boolean expression ? value if true : value if false

Conditional Operator

- If grade is greater than or equal to 65 then the string "pass" is placed in the message variable otherwise the string "fail" is placed in the message variable.

```
public static void main(String [] args)
{
   String message;
   int grade = 55;

   message = (grade >= 65) ? "pass" : "fail";
   System.out.println(message);
}
```

Conditional Operator

Conditional Operator

- You can use complex expression inside of the conditional
- In this example, if the customer buys 5 or more items she gets a cheaper price

```
public static void main(String [] args)
{
   int items=10;
   double cost;

   cost = (items >= 5) ? 1.50 * items : 2.00 * items;
   System.out.printf("Cost = %f\n", cost);
}
```

While Loop

- While is a repetition statement – A loop.
- The purpose of a loop is to repeatedly do something.
- While is a *pretest* loop – The test is at the beginning of the loop.
- If the boolean expression is true then do whatever is inside the loop body.
- For example:

```
While (boolean expression is true)
      do the body of loop
```

- How can we print a message 5 times.
- One way is to put 5 calls to print the message one after the other.
- For example:

```
public static void main(String [] args)
{
   System.out.printf("Yanks");
   System.out.printf("Yanks");
   System.out.printf("Yanks");
   System.out.printf("Yanks");
   System.out.printf("Yanks");
}
```

- Now use a while loop to do this…

While Loop

- Print a message 5 times using a loop.
- For example:

```
public static void main(String [] args)
{
   int num=1;

   while (num <= 5) // Keep executing loop body while true
   {
         System.out.println("Yanks");
         num =  num + 1;          // Go to the next number
   }
}
```

While Loop

- Will this work? If not what is the problem?

```
public static void main(String [] args)
{
   int num=1;

   while (num <= 5) // Keep executing loop body while true
   {
         System.out.println("Yanks");
   }
}
```

While Loop

While Loop

- If we leave out "num = num + 1" then the loop will never stop repeating.
- That would be an example of an infinite loop.

```
public static void main(String [] args)
{
    int num=1;

    while (num <= 5) // Keep executing loop body while true
    {
          System.out.println("Yanks");
          num = num + 1;       // This adds to num and moves
                               // the program towards the
                               // stopping condition of the loop.
    }
}
```

- When programming a loop you must make sure that the stopping condition of the loop will eventually occur.

- If a loop keeps repeating forever then it is an "infinite loop".

While Loop

- Next example.
- Write a loop that will keep adding 2 to a total up until a certain point.
- So, keep adding 2 to the total variable while the value of total is less than or equal to 4.
- Here is the code…

While Loop

- The following while loop keeps adding 2 to the value of total while total is less than or equal to 4.

```
public static void main(String [] args)
{
        int total = 0;

        while (total <= 4)
                total = total + 2;

        System.out.printf("Total = %d\n", total);
}
```

What is the final value of total?

While Loop

- While is similar to if in that only the first statement after the test condition is part of the loop body.
- Suppose I want to modify the previous program so that it will print the value of total each time through the loop.
- How do I update the program to make it happen?
- Will the following work?

While Loop

While Loop

- Will this print total each time through the loop?

```
public static void main(String [] args)
{
        int total = 0;

        while (total <= 4)
                total = total + 2;
                System.out.printf("Total = %d\n", total);
}
```

While Loop

- ***No***. It will not print total each time through the loop.
- You need to put the loop statements in braces if you want more than one statement in the loop body (same as with an if statement).

```
public static void main(String [] args)
{
        int total = 0;

        while (total <= 4)
        {                           // Opening Brace
                total = total + 2;
                System.out.printf("Total = %d\n", total);
        }                           // Closing Brace
}
```

- Now write a program that will print the numbers 1 through 5 on the screen without using a loop.
- Here is the simple version…

While Loop

- Another example.
- Print all of the numbers from 1 to 5 (no loop).

```
public static void main(String [] args)
{
        System.out.println(1);
        System.out.println(2);
        System.out.println(3);
        System.out.println(4);
        System.out.println(5);
}
```

- Now do this using a variable…

While Loop

- Another example.
- Print all of the numbers from 1 to 5 (no loop).

- Will this work?

```
public static void main(String [] args)
{
        int num = 1;    // Start num at 1
        System.out.println(num);
        System.out.println(num);
        System.out.println(num);
        System.out.println(num);
        System.out.println(num);
}
```

While Loop

- ***No***. You need to go to the next number each time before you print it.
- Here is the correct way to do it.

```
public static void main(String [] args)
{
        int num = 1;    // Start num at 1
        System.out.println(num);
        num = num + 1;   // Go to the next number
        System.out.println(num);
        num = num + 1;   // Go to the next number
        System.out.println(num);
        num = num + 1;   // Go to the next number
        System.out.println(num);
        num = num + 1;   // Go to the next number
        System.out.println(num);
}
```

While Loop

- Now write a program that will print the numbers 1 through 5 on the screen ***using*** a loop.
- Here is the loop version…

While Loop

While Loop

- Another example.
- Print all of the numbers from 1 to 5 using a loop.

```
public static void main(String [] args)
{
        int num = 1;    // Start num at 1

        while (num <= 5)   // Keep going while num <= 5
        {
                System.out.println(num);
                num = num + 1;   // Go to the next number
        }
}
```

- Now write a program that will add all of the numbers from numbers 1 to 5.
- Here is the non-loop version…

While Loop

While Loop

- Add all of the numbers from 1 to 5 (no loop).

```
public static void main(String [] args)
{
        int num = 1, total = 0;   // Start num at 1, total at 0
        total = total + num; // Add num to total
        num = num + 1;       // Go to the next number
        total = total + num; // Add num to total
        num = num + 1;       // Go to the next number
        total = total + num; // Add num to total
        num = num + 1;       // Go to the next number
        total = total + num; // Add num to total
        num = num + 1;       // Go to the next number
        total = total + num; // Add num to total
}
```

- Add all of the numbers from 1 to 5 using a loop.

```
public static void main(String [] args)
{
        int num = 1, total = 0; // Start num at 1, total at 0

        while (num <= 5)   // Keep going while num <= 10
        {
                total = total + num;  // Add num to total
                num = num + 1;        // Go to the next num
        }

        System.out.println(total);
}
```

While Loop

Compound Assignment Operators

- Some operators allow a shorthand version.
- For example:

  total = total + 5;

  Can be written as:

  total += 5;

- The += operator is called the "addition compound assignment operator".
- Here are some other compound assignment operators…

- total = total – 5;
  total -= 5;

- Total = total * 5;
  total *= 5;

- total = total / 5;
  total /= 5;

- total = total % 5;
  total %= 5;

Compound Assignment Operators

- Add 1 to a variable
  or
- Subtract 1 from a value

- For example:

```
Num = num + 1;
num++;

Num = num – 1;
num--;
```

Increment and Decrement Operators

- Preincrement operator:
  `++num;`
- Postincrement operator:
  `num++;`
- Both add 1 to num.
- Difference occurs when they appear in an expression.

Preincrement Vs Postincrement

- Example
- What is the value that gets printed?

Hint: You must do the increment ***before*** the expression is evaluated.

```
public static void main(String [] args)
{
  int num, c = 1;

  num = ++c;      // preincrement
  System.out.println(num);
}
```

Preincrement Vs Postincrement

- 2 gets printed

- The value inside of c is incremented then the right hand side can be evaluated.

```
public static void main(String [] args)
{
   int num, c = 1;

   num = ++c;       // preincrement
   System.out.println(num);
}
```

What if we use the postincrement operator...

Preincrement Vs Postincrement

- This example uses the postincrement operator.

```
public static void main(String [] args)
{
  int num, c = 1;

  num = c++;      // postincrement
  System.out.println(num);
}
```

Now what value gets printed?

Preincrement Vs Postincrement

Preincrement Vs Postincrement

- 1 gets printed this time!!!
- The expression is evaluated with the original value of c.
- After the expression is evaluated and just before the computer moves on to the next instruction the value of c is incremented.

```
public static void main(String [] args)
{
  int num, c = 1;

  num = c++;       // postincrement
  System.out.println(num++);
}
```

- The pre and post versions of the decrement operator behave the same as their increment counterparts.
- --num   predecrement operator
  First decrement then evaluate expression
- Num– postdecrement operator
  Evaluate expression the decrement

Predecrement Vs Postdecrement

- Here is the program to add the numbers 1 to 5 using compound and increment operators.

```
public static void main(String [] args)
{
        int num = 1, total = 0; // Start num at 1, total at 0

        while (num <= 5)   // Keep going while num <= 10
        {
                total += num;         // Compound assignment
                num++;                // Increment operator
        }

        System.out.println(total);
}
```

Compound and Increment

- End of Slides

End of Slides