Introduction to Java
Notes Packet #2

Name: _____

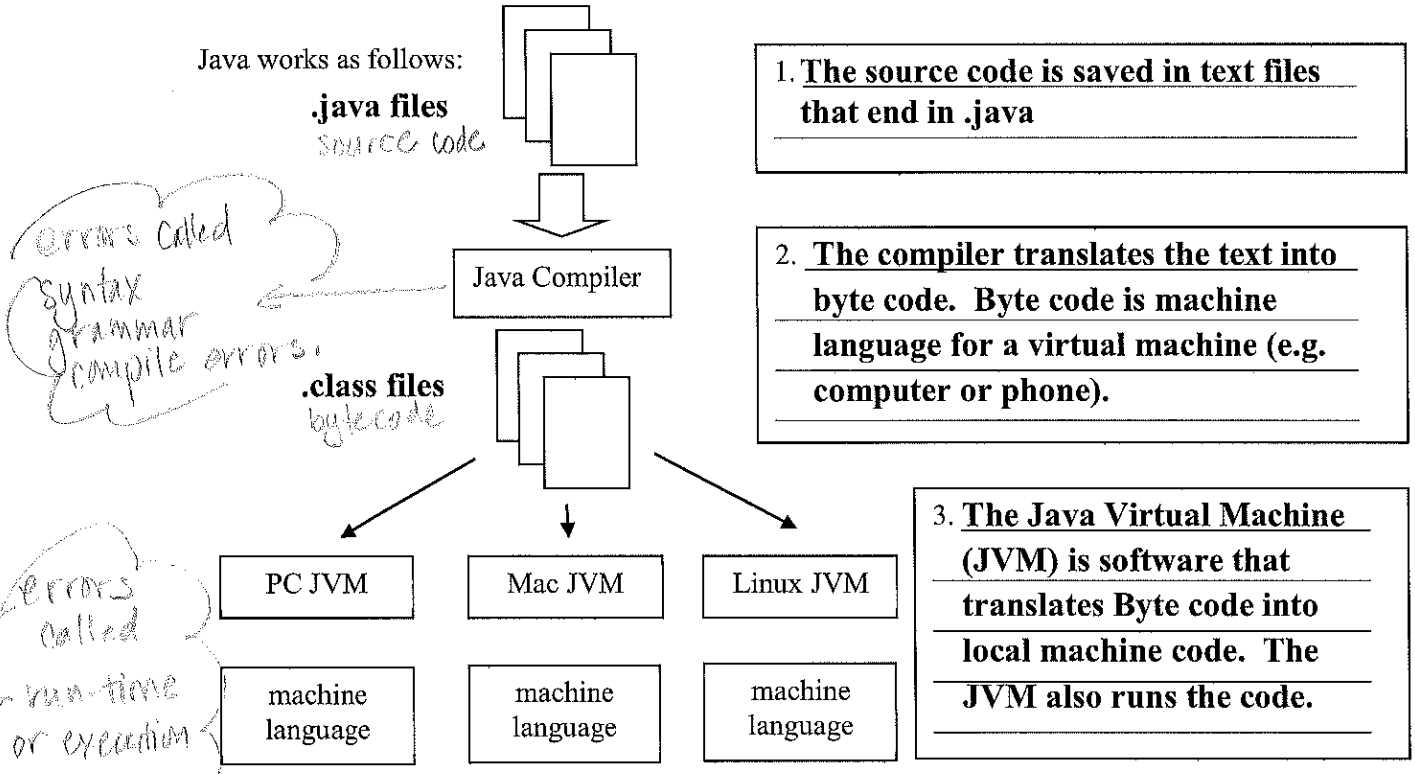
Objectives: By the completion of this packet, students should be able to

- describe the difference between .java and .class files and the JVM.
- create and use flow charts
- evaluate and use Boolean expressions
- create and use if and if-else statements
- use the && (AND) and || (OR) operators
- use some methods of the Math class.

Compiling Code in Java

A compiler is a program that translates code to machine language that the central processing unit (the computer's "brain") can understand. In many languages, the source code must be customized for different operating systems and then compiled for certain types of processors.

Java works as follows:



1. **The source code is saved in text files that end in .java**

2. **The compiler translates the text into byte code. Byte code is machine language for a virtual machine (e.g. computer or phone).**

3. **The Java Virtual Machine (JVM) is software that translates Byte code into local machine code. The JVM also runs the code.**

errors called syntax grammar compile errors.

errors called run-time or execution errors

Programmers need the compiler and JVM to develop software. A finished program is delivered as a collection of class files. End users only need the JVM to run the class files.

if results come out but are wrong, there are logic or semantic errors!

Boolean Expressions

A Boolean expression is an expression that is either **true** or **false**.

For example: x >= 8

The above expression compares x to 8. If x is greater than or equal to 8, then the expression is true. Otherwise the expression is false. in coding, boolean expressions are for questioning or branching.

Java uses 6 relational operators in its Boolean expressions.

> < >= <= == != (means not equal)

1. If *x* is an int with a value of 12, what is the value of this expression? **true**	x > 10
2. If *num* is an int with a value of 22, what is the value of this expression? **false**	num == 14
3. If *y* is an int with a value of 7, what is the value of this expression? **false**	50 <= y better: y >= 50

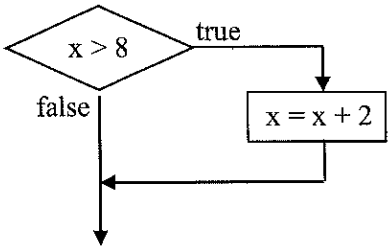
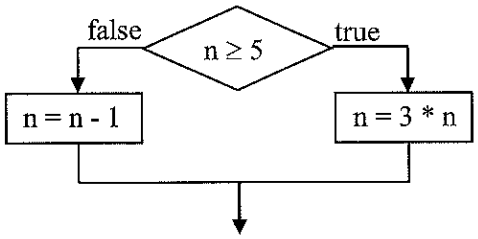
Boolean expressions can be manipulated the same way inequalities in math can be manipulated to produce equivalent expressions. Rewrite the expression so that all x terms are combined on the left and the coefficient is 1.	x > 20 + 5 * x **-5 *x - 5* x** **-4 * x > 20** **-4 * x > 20** **-4 -4** **x < -5**

4. Are the two expressions equivalent? **no**	num <= k	k <= num
5. Are the two expressions equivalent? **yes**	h != 20	20 != h
6. Are the two expressions equivalent? **yes**	20 - y > 5	y < 15

use simplest form when coding

The if and if-else Statements

These statements are sometimes called control statements because they control the flow of the program. Some texts refer to them as conditionals because they represent a condition that may be true or false.

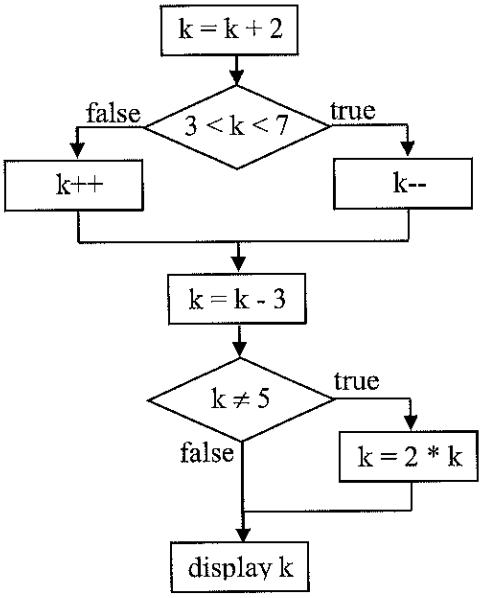
Flowchart	Java Code
	**if (x > 8) {** **x = x + 2;** **}** **The boolean expression must be enclosed by parentheses. The curly braces define the block of code that should be executed or skipped.** a block is 1 or more lines of code
	**if (n >= 5) {** **n = 3 * n;** **} else {** **n = n - 1;** **}**

Logical Operators. More complicated Boolean expressions can be composed through the use of logical operators such as AND and OR.

The following table shows how certain expressions can be written.

English	Mathematically	Java
If the number is greater than 0 and less than or equal to 5.	**0 < x ≤ 5**	**x > 0 && x <= 5** **The expressions to the left and right of && must be complete boolean expressions.**
If the number is less than 12 or greater than 65	**x< 12 or x>65**	**x < 12 || x > 65**

Example. Complete the code to the right based on the flowchart below.




```
import java.util.Scanner;

public class Example {
        public static void main(String[] args) {
                Scanner x = new Scanner( System.in );
                System.out.print("Enter an integer ");
                int k = x.nextInt();

  k = k + 2;
  if ( k > 3 && k < 7 )
        k--;
  else
        k++;


  k = k - 3;
  if ( k != 5 )
        k = 2 * k;


  System.out.println( k );


        }
}
```

The curly braces {} are not required if there is only one statement associated with the *if* or *else*.

may use just not nec.

The Math Class.

The Math Class contains a collection of standard math functions. Here are some examples on how to use them.

To take the square root of a number:

```
double a = Math.sqrt( 16 );        // a equals 4.0
a = Math.sqrt( a );                // a equals 2.0
```

To raise a number to nth power:

```
double b = Math.pow( 2, 3 );       // equivalent to 2^3, b equals 8.0
double c = Math.pow( b, 2.0 );     // c equals 64.0
```

Note: to square a number you can always multiply it by itself.

```
int e = 7;
int f = e * e;                     // or int f = (int) Math.pow( e, 2 );
```

show java api Math class here

Find the absolute value:

```
int x = Math.abs( -5 );          // if the argument is an int, the method returns an int
double y = Math.abs( -7.3 );     // if the argument is a double, the method returns a double
```

To generate a random decimal between 0 (inclusive) and 1 (exclusive):

```
double g = Math.random();        // 0.0 ≤ g < 1.0
double h = 5 * Math.random();    // 0.0 ≤ h < 5.0
```

To generate a random integer from min (inclusive) and max (inclusive), use the following formula:

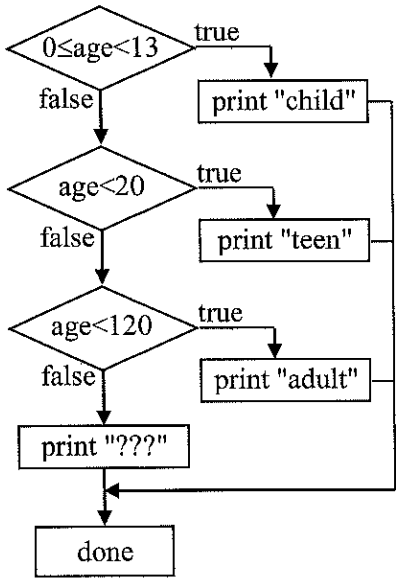
```
int num = (int) ( range * Math.random() ) + min;     // where range = max – min + 1
```

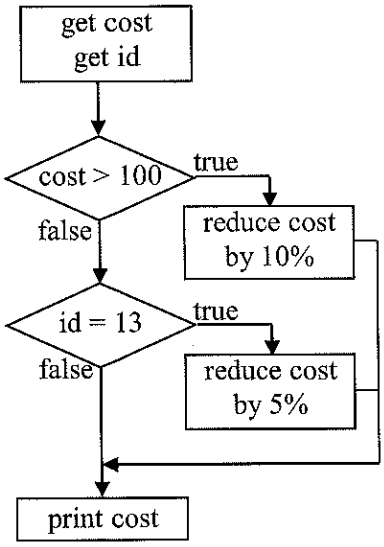
For example, to generate a random integer between 1 and 6:

```
int num = (int) ( 6 * Math.random() ) + 1;           // num is equal to 1, 2, 3, 4, 5, or 6
```

1. *a* is a random decimal within these limits **-3.0** ≤ a < **7.0**	double a = 10 * Math.random() - 3;
2. *b* is a random integer within these limits **3** ≤ b ≤ **7**	int b = (int)(5 * Math.random()) + 3;
3. *c* is a random integer within these limits **0** ≤ c ≤ **11**	int c = (int)(12 * Math.random());
4. *d* is a random integer within these limits **-7** ≤ d ≤ **-5**	int d = (int)(3 * Math.random()) - 7;

if-else if Statements. The basic if - else structure can be expanded to handle multiple conditionals through the use of else-if statements. For example, if you want to determine if a grade is an A, B, C, D, or E, a simple if - else statement won't work.

Flowchart	Java Code
	**if (age >= 0 && age < 13)** **System.out.println("child");** **else if (age < 20)** **System.out.println("teen");** **else if (age < 120)** **System.out.println("adult");** **else** **System.out.println("???");**

	import java.util.Scanner;
	
	public class Example {
	public static void main(String[] args) {
	Scanner sc = new Scanner(System.in);
	System.out.print("Enter cost ");
	double cost = sc.nextDouble();
	System.out.print("Enter id ");
	int id = sc.nextInt();
	if (cost > 100)
	cost = 0.9 * cost;
	else if (id == 13)
	cost = 0.95 * cost;
	
	System.out.println(cost);
	}
	}

IMPORTANT. **Only the statements associated with the <u>first</u> true expression are executed. The others are ignored.**