Introduction to JAVA

Subject Code: 333514(33)

Dr. Lokesh Kumar Sharma

Ph.D. (CS), DAAD Fellow Germany

Reader

Department of IT and MCA

drlokeshksharma@gmail.com

Syllabus

3. Operating System Design & Implementation, Tanenbaum, A. S., Prectice Hall NJ

CHHATTISGARH SWAMI VIVEKANAND TECHNICAL UNIVERSITY
BHILAI (C.G.)

Semester: V
Subject: Introduction To JAVA
Total Theory Periods: 40
Total Marks in end semester examination: 80
Minimum number of class tests to be conducted: 02

Branch: Information Technology.
Code: 333514(33)
Total Tutorial Periods: 12

UNIT – I **INTRODUCTION & FUNDAMENTALS**
Features of Java, newly added features in Java2, introduction to OOPS, data types, variables, literals, expressions, operators, arrays and programming constructs, Garbage Collection, Comparision with C++, Java Virtual Machines, Java Class Libraries, JIT, Overview of Java Technologies: Applets, Beans, RMI, Servlets, JSP, JSF, CORBA .

Unit – II **CLASSES AND OBJECTS**
Classes and Objects, Objects and References, , Method: Defining method, calling method, passing arguments to method, this keyword, overloading method, static, Access specifiers; public, default, private & protected. Command line arguments, constructors and finalizers, overloading constructors, inner classes.
Introduction to inheritance; definition and advantages, overriding, Super , final and abstract classes, Interface , Package.

Unit – III **EXCEPTIONS, STRING AND VECTOR**
Basics of exception handling, default Exception handling, try and catch, Multiple catch statements, try-catch- finally, uses of throw and throws, Strings: string constructor, string arithmetic, string methods, stringbuffer and methods, Introduction and programming using Vector, Iterator and Enumeration

Unit – IV **MULTITHREADING**
Thread Concepts, Thread lifecycle, Runnable Vs Thread Class, Thread Priority, Thread Methods, Thread Synchronization: Concept of Monitor, Synchronized methods & Synchronized blocks.

Unit –V **INTERNET PROGRAMMING WITH JAVA**
AWT, applets and application, user interfacing components,Events and Event Handling, Overview of Swing Components, Java Database Connectivity: JDBC, ODBC, executing DDL, DML commands, statement, prepared statement and callable statement, Java Stored Procedures.

Text books:
1. Java complete reference – Herbert Schildt (TMH)
2. Java how to program – Dietel and Dietel

Reference books:
1. Programming with Java :- Schaum's series
2. Java 2 Black book – Steven Holzner
3. Java Examples in a nutshell – O' Reilly
4. Core Java – Cay S. Horstman, Gary Cornell

Java

How to find, install and begin to use the Java programming language.

History

James Gosling Circa 1971

James Gosling Today

- James Gosling began developing Java beginning in 1991
- It was first called “Project Green” and ‘Oak”
- First developed for remote cable TV boxes
- Sun Microsystems released to a select group on the Web site wicked.neato.org as Java 1.0 in 1995
- James Naughton creates “HotJava” in 1995. it’s a web browser that lets you run “Applets”. The entire browser is written in Java.
- Java currently released version JDK7 or Java SE 7.0

History...

1. JDK 1.0 (January 23, 1996)
2. JDK 1.1 (February 19, 1997)
3. J2SE 1.2 (December 8, 1998)
4. J2SE 1.3 (May 8, 2000)
5. J2SE 1.4 (February 6, 2002)
6. J2SE 5.0 (September 30, 2004)
7. Java SE 6 (December 11, 2006)

Version of java

- Java Language vs Java Platform
  - Current version of the *language* is JKD7
  - Core language plus additional APIs is called the Java 2 *platform*
  - Three versions of the Java 2 Platform, targeted at different uses
- Java 2 Micro Edition (J2ME)
  - Very small Java environment for smart cards, pages, phones, and set-top boxes
  - Subset of the standard Java libraries aimed at limited size and processing power
- Java 2 Standard Edition (J2SE)
  - The basic platform, which this course will cover
- Java 2 Enterprise Edition (J2EE)
  - For business applications, web services, mission-critical systems
  - Transaction processing, databases, distribution, replication

Java IDE Tools

- Forte by Sun MicroSystems
- Borland Jbuilder
- Microsoft Visual J++
- WebGain Café
- IBM Visual Age for Java
- Eclipes

Download Java

You want a JDK, not just a JRE

Create a “Temp” directory on your PC or laptop

Go to http://java.sun.com

Go to the “Popular Downloads” section and select “Java SE”

Select a JDK without Netbeans (We’ll talk about this later)

Agree to the accept the use policy

Right Click and Save the Offline Windows JDK to your temp dir

Help on the Web

http://java.sun.com/javase/6/docs/api/

http://www.javaranch.com

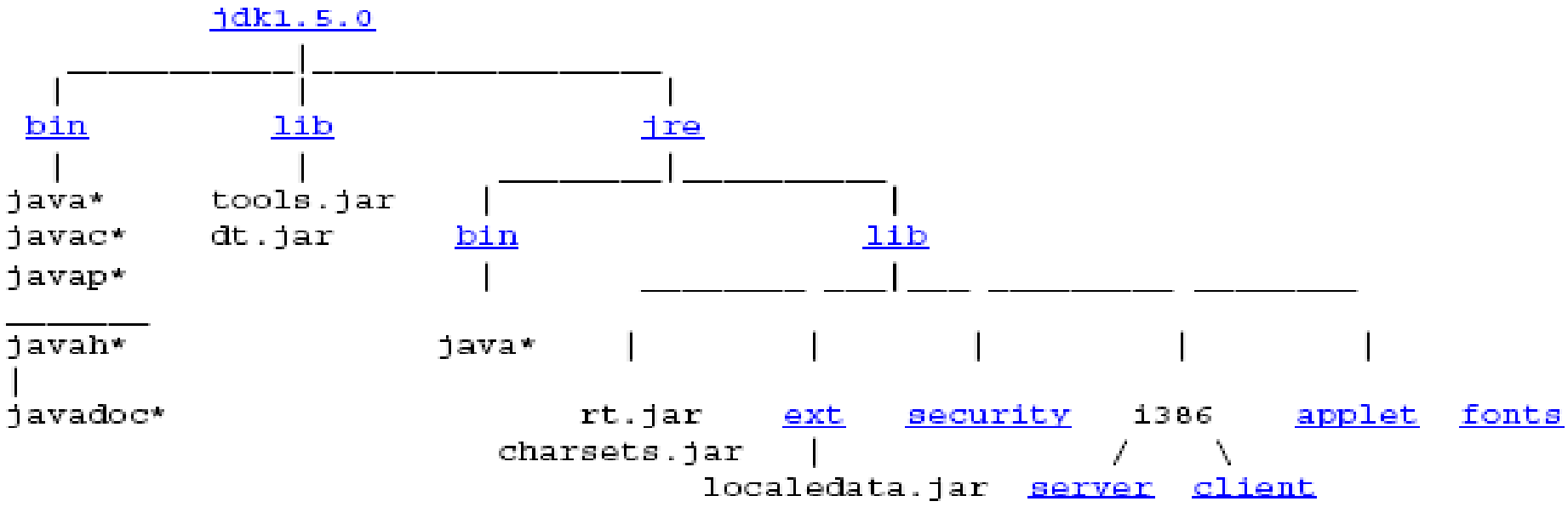
http://java.sun.com/docs/books/tutorial/

http://www.sorcon.com/java2/

http://remus.rutgers.edu/freestuff

JDK Directory Structure

Assuming the JDK software is installed at /jdk1.5.0, here are some of the most important directories:



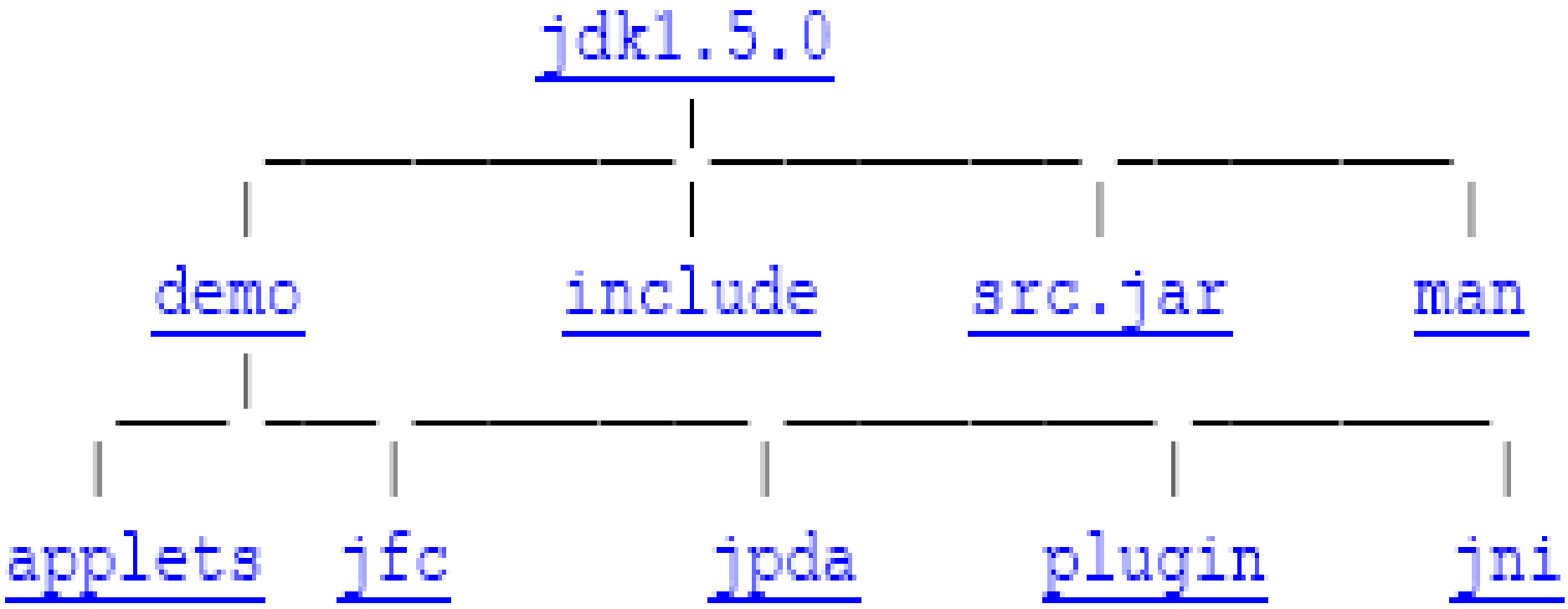
JDK Directory Structure....

Directory	Discription
/jdk1.5.0	The root directory of the JDK software installation. Contains copyright, license, and README files. Also contains src.jar, the archive of source code for the Java 2 platform.
/jdk1.5.0/bin	The executables for all the development tools contained in the Java 2 JDK. The **PATH** environment variable should contain an entry for this directory. For more information on the tools
/jdk1.5.0/lib	Files used by the development tools. Includes tools.jar, which contains non-core classes for support of the tools and utilities in the JDK
/jdk1.5.0/jre	The root directory of the Java runtime environment used by the JDK development tools. The runtime environment is an implementation of the Java 2 platform. This is the directory referred to by the java.home system property.
/jdk1.5.0/jre/bin	Executable files for tools and libraries used by the Java platform. The executable files are identical to files in /jdk1.5.0/bin. The **java** launcher tool serves as an application launcher, in place of the old **jre** tool that shipped with 1.1 versions of the JDK software. This directory does not need to be in the **PATH** environment variable.
/jdk1.5.0/jre/lib	Code libraries, property settings, and resource files used by the Java runtime environment

JDK Directory Structure....

Directory	Discription
/jdk1.5.0/jre/lib/ext	Default installation directory for Extensions to the Java platform. This is where the JavaHelp jar file goes when it is installed
/jdk1.5.0/jre/lib/security	Contains files used for security management. These include the security policy (java.policy) and security properties (java.security) files
/jdk1.5.0/jre/lib/i386/client	Contains the .so file used by the Java HotSpot Client Virtual Machine, which is implemented with Java HotSpot™ technology. This is the default VM.
/jdk1.5.0/jre/lib/i386/server	Contains the .so file used by the Java HotSpot Server Virtual Machine.
/jdk1.5.0/jre/lib/applet	Jar files containing support classes for applets can be placed in the lib/applet/ directory
/jdk1.5.0/jre/lib/fonts	Font files for use by platform.

Additional Files and Directories



Additional Files and Directories…

Directory	Discription
/jdk1.5.0/demo	Examples, with source code, that show you how to program for the Java platform.
/jdk1.5.0/demo/applets	Applets that can be used on a Web page
/jdk1.5.0/demo/jfc	Examples that use Java 2D™ and JFC/Swing components.
/jdk1.5.0/src.jar	Archive containing source code for the Java 2 platform.
/jdk1.5.0/include	C-language header files that support native-code programming using the Java Native Interface and the Java Virtual Machine Debugger Interface.
/jdk1.5.0/demo/jni	Example classes and C code that demonstrate access to poll(2) functionality from the Java 2 Platform.

Java Basic Tools

Tool Name	Brief Description
javac	The compiler for the Java programming language.
java	The launcher for Java applications.
javadoc	API documentation generator.
appletviewer	Run and debug applets without a web browser.
jar	Create and manage Java Archive (JAR) files.
jdb	The Java Debugger.
javah	C header and stub generator. Used to write native methods.
javap	Class file disassembler
extcheck	Utility to detect Jar conflicts.

Definition of Java (As per Sun MicroSystem)

- **A simple, Object Oriented, distributed, interpreted, robust, secure, architecture, neutral, protable, high performance, mutithread, and dynamic language.**

Characterstics of Java

- Architecture Neutral
- Object Oriented Programming
- Simple
- Distributed
- Interpreted and complied
- Secure
- Portable
- Web Enable
- Multi-Threading
- Garbage Collection

Characterstics of Java...

- **Simple**
  - fixes some clumsy features of C++
  - no pointers
  - automatic garbage collection
  - rich pre-defined class library
- **Object oriented**
  - focus on the data (objects) and methods manipulating the data
  - all functions are associated with objects
  - almost all datatypes are objects (files, strings, etc.)
  - potentially better code organization and reuse

Characterstics of Java...

- **Interpreted**
  - java compiler generate byte-codes, not native machine code
  - the compiled byte-codes are platform-independent
  - java bytecodes are translated on the fly to machine readable instructions in runtime (Java Virtual Machine)
- **Portable**
  - same application runs on all platforms
  - the sizes of the primitive data types are always the same
  - the libraries define portable interfaces

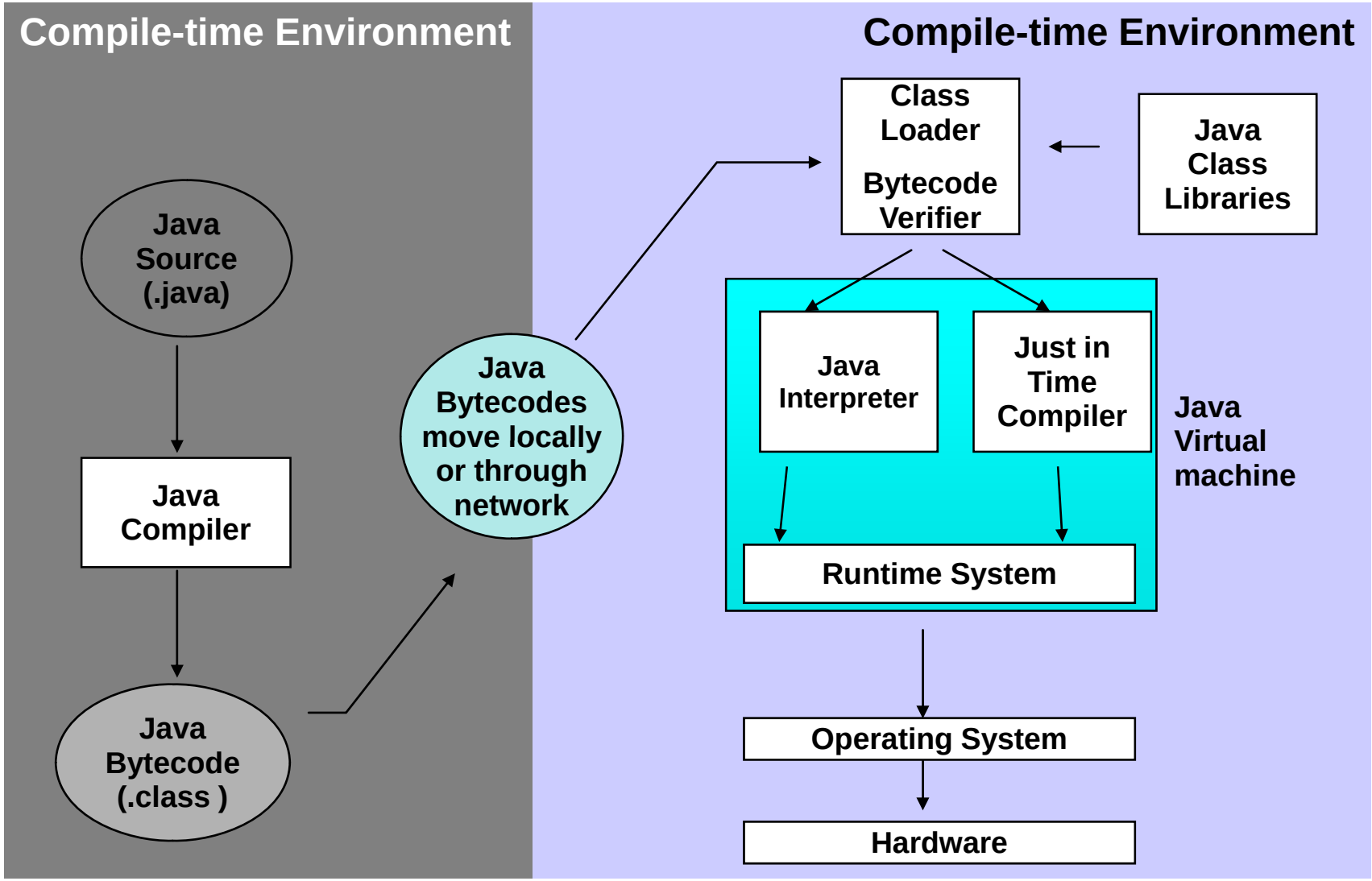
Characterstics of Java

- **Reliable**
  - extensive compile-time and runtime error checking
  - no pointers but real arrays. Memory corruptions or unauthorized memory accesses are impossible
  - automatic garbage collection tracks objects usage over time
- **Secure**
  - usage in networked environments requires more security
  - memory allocation model is a major defense
  - access restrictions are forced (private, public)

Characterstics of Java

- **Multithreaded**
  - multiple concurrent threads of executions can run simultaneously
  - utilizes a sophisticated set of synchronization primitives (based on monitors and condition variables paradigm) to achieve this
- **Dynamic**
  - java is designed to adapt to evolving environment
  - libraries can freely add new methods and instance variables without any effect on their clients
  - interfaces promote flexibility and reusability in code by specifying a set of methods an object can perform, but leaves open how these methods should be implemented
  - can check the class type in runtime

How it works…!


Compile-time Environment
Compile-time Environment
Class Loader
Bytecode Verifier
Java Class Libraries
Java Source (.java)
Java Compiler
Java Bytecode (.class)
Java Bytecodes move locally or through network
Java Interpreter
Just in Time Compiler
Java Virtual machine
Runtime System
Operating System
Hardware

Simple Java Program

```
public class Hello
{
    public static void main (String args[])
    {
     System.out.println("This is first java program");
    }
}
```

How to compile and run java program

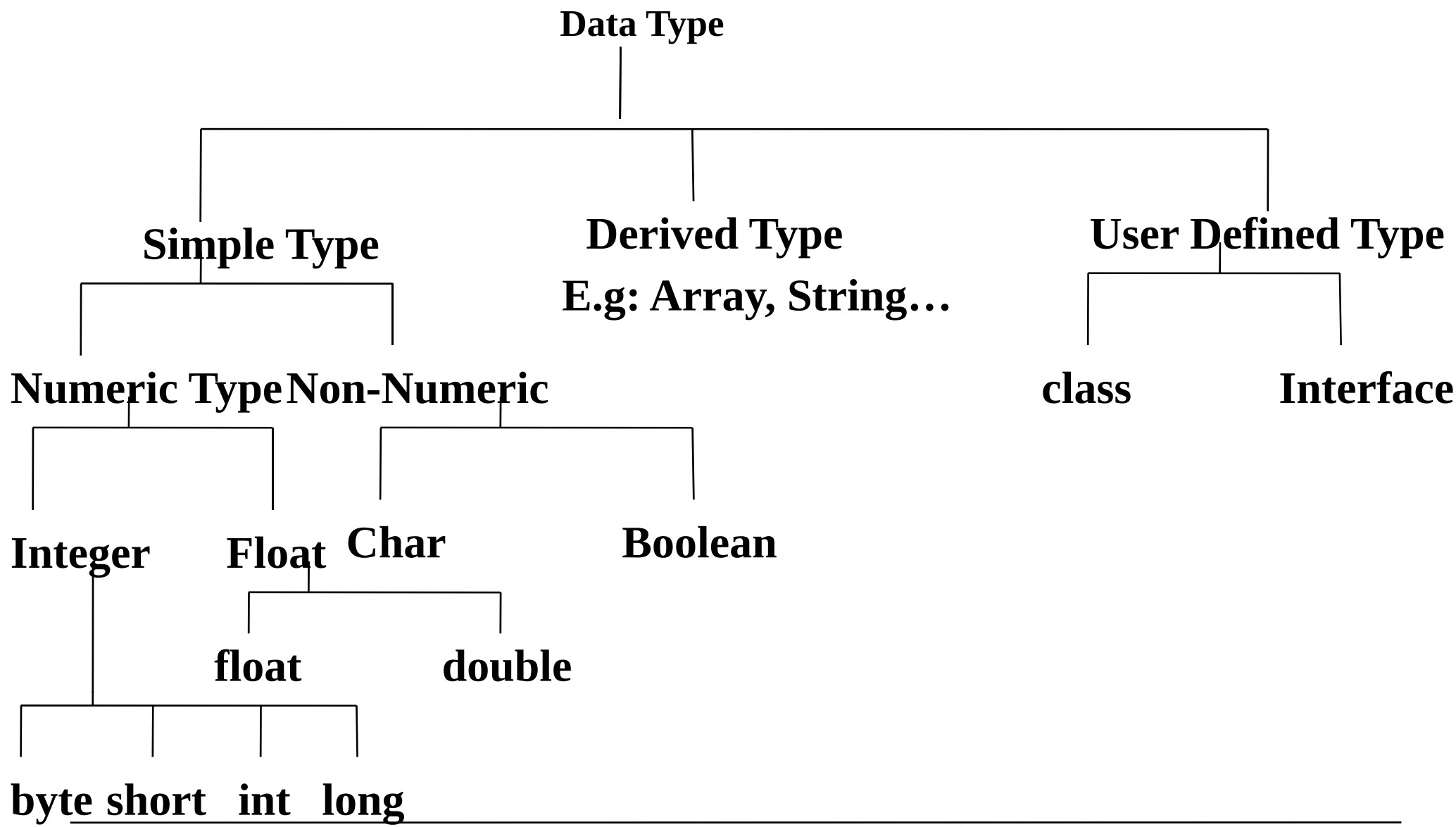
- Save java program with extension .java
- Compile java program
  - javac Hello.java
- Run java program
  - java Hello

Java Keywods/Reserved Words

- *Reserved words* or *keywords* are words that have a specific meaning to the compiler and cannot be used for other purposes in the program. For example, when the compiler sees the word class, it understands that the word after class is the name for the class. Other reserved words are

abstract	continue	goto	package	synchronized
assert	default	if	private	this
boolean	do	implements	protected	throw
break	double	import	public	throws
byte	else	instanceof	return	transient
case	extends	int	short	try
catch	final	interface	static	void
char	finally	long	strictfp	volatile
class	float	native	super	while
const	for	new	switch	

Data Type


Data Type
Simple Type
Derived Type
E.g: Array, String…
User Defined Type
Numeric Type
Non-Numeric
class
Interface
Integer
Float
Char
Boolean
float
double
byte
short
int
long

Data Type

Data Type	Initial Value	Memory Required	Range
byte	0	1 Byte	–128 to 127
short	0	2 Byte	–32,768 to 32,767
int	0	4 Byte	2,147,483,648 to 2,147,483,647
long	0L	8 Byte	–9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
float	0.0f	4 Byte	1.4e−045 to 3.4e+038
double	0.0	8 Byte	4.9e–324 to 1.8e+308
char		2 Byte	0 to 65,536
boolean			true or false

Literals

```
boolean result = true;
byte b = 100;
short s =10000;
 int i = 100000;
```

• **int**:
– int decVal = 26; // The number 26, in decimal
– int hexVal = 0x1a; // The number 26, in hexadecimal
-int octVal = 010; // The number 8 , in octal

• **float and double**:
– E or e (for scientific notation)
– F or f (32-bit float literal)
– D or d (64-bit double literal).
– double d1 = 123.4;
– double d2 = 1.234e2; // same value as d1, but in scientific notation
– float f1 = 123.4f;

Character in Java

- char ch='\u0108‘;
- char t = ‘C‘;

Escape Sequence Character

Escape Sequence	Description
\ddd	Octal character (ddd)
\uxxxx	Hexadecimal UNICODE character (xxxx)
\'	Single quote
\"	Double quote
\\	Backslash
\r	Carriage return
\n	New line (also known as line feed)
\f	Form feed
\t	Tab
\b	Backspace

Comments in Java

- Comments are simply notes for the programmer to remind themseleves and other programmers what a program is meant to do. Comments can contain any text you like

```
// this is a single line comment
// and here is another

/* this comment can
   extend over several lines.......   */

/** This comment is document comment. It uses to
  incorporate yours line with java help file. ...*/
```

- Comments are not java commands. **Always** comment your programs! It helps you remember what's going on.

Operators

Arithmetic Operators(1)

Operator	Result
+	Addition
–	Subtraction
*	Multiplication
/	Division
%	Modulus

Example

```
class Modulus
 {
       public static void main(String args[])
       {
         int x = 42;
         double y = 42.25;
         System.out.println("x mod 10 = " + x % 10);
        System.out.println("y mod 10 = " + y % 10);
      }
}
```

Arithmetic Operators(2)

++	Increment
+=	Addition assignment
–=	Subtraction assignment
*=	Multiplication assignment
/=	Division assignment
%=	Modulus assignment
– –	Decrement

Example:

```
class IncDec
{
        public static void main(String args[])
      {
         int a = 1;
         int b = 2;
         int c,d;
        c = ++b;
        d = a++;
        c++;
        System.out.println("a = " + a);
        System.out.println("b = " + b);
        System.out.println("c = " + c);
        System.out.println("d = " + d);
}
}
```

```
class OpEquals
{
        public static void main(String args[])
       {
         int a = 1;
         int b = 2;
         int c = 3;
          a += 5;
          b *= 4;
          c += a * b;
          c %= 6;
          System.out.println("a = " + a);
         System.out.println("b = " + b);
         System.out.println("c = " + c);
       }
}
```

Bitwise Operators(1)

Operator	Result
~	Bitwise unary NOT
&	Bitwise AND
|	Bitwise OR
^	Bitwise exclusive OR
>>	Shift right
>>>	Shift right zero fill
<<	Shift left

Bitwise Operators(2)

Operator	Result
&=	Bitwise AND assignment
|=	Bitwise OR assignment
^=	Bitwise exclusive OR assignment
>>=	Shift right assignment
>>>=	Shift right zero fill assignment
<<=	Shift left assignment

Relational Operators

Operator	Result
==	Equal to
!=	Not equal to
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to

Example:

```
public class RelationalOperatorsDemo
{
   public static void main(String args[])
    {
       int x=10,y=5;
       System.out.println("x>y:"+(x>y));
       System.out.println("x<y:"+(x<y));
       System.out.println("x>=y:"+(x>=y));
       System.out.println("x<=y:"+(x<=y));
       System.out.println("x==y:"+(x==y));
       System.out.println("x!=y:"+(x!=y));
    }
}
```

Boolean Logical Operators(1)

Operator	Result
&	Logical AND
|	Logical OR
^	Logical XOR (exclusive OR)
||	Short-circuit OR
&&	Short-circuit AND
!	Logical unary NOT

Boolean Logical Operators(2)

Operator	Result
&=	AND assignment
|=	OR assignment
^=	XOR assignment
==	Equal to
!=	Not equal to
?:	Ternary if-then-else

```
public class TernaryOperatorDemo
{
  public static void main(String args[])
   {
      int x=10,y=12;
      int z;
      z= x > y ? x : y;
      System.out.println("Z="+z);
}
```

Precedence Of The Operators(1)

( )	[ ]	.
++	– –	~ !
*	/	%
+	–	
>>	>>>	<<
> >=	< <=	

Precedence Of The Operators(2)

== !=

&

^

|

&&

||

?:

= op=

Control Flow Statements

- Java executes one statement after the other in the order they are written

- Many Java statements are flow control statements:

  1. Alternation: if, if else, switch

  2. Looping: for, while, do while

  3. Escapes: break, continue, return

If – The Conditional Statement

- The if statement evaluates an expression and if that evaluation is true then the specified action is taken

```
if ( x < 10 ) x = 10;
```

- If the value of x is less than 10, make x equal to 10
- It could have been written:

```
if ( x < 10 )
x = 10;
```

- Or, alternatively:

```
if ( x < 10 ) { x = 10; }
```

If... else

- The if … else statement evaluates an expression and performs one action if that evaluation is true or a different action if it is false.

```
if (x != oldx) {
   System.out.print("x was changed");
 }
 else {
   System.out.print("x is unchanged");
 }
```

Nested if … else

```
if ( A > B ) {
        if ( A>C) {
         System.out.println( “A is Largest”);
        }
   else {
         System.out.println(“C is Largest”);
        }
}
else if (B>C)
{
   System.out.print(“B is Largest”);
}
else
    {
        System.out.println(“C is Largest“);
}
```

The switch Statement

```
switch ( n ) {
  case 1:
   // execute code block #1
   break;
  case 2:
   // execute code block #2
   break;
   default:
   // if all previous tests fail then
   //execute code block #4
   break;
}
```

The for loop

- Loop n times

```
for ( i = 0; i < n; n++ ) {
  // this code body will execute n times
  // i from  0 to n-1
}
```

- Nested for:

```
for ( j = 0; j < 10; j++ ) {
  for ( i = 0; i < 20; i++ ){
   // this code body will execute 200 times
  }
}
```

while loops

```
int i =1;
while(i<= 10) {
  System.out.print( “I =” + i);
  i++;

}
```

do {… } while loops

```
int i =1;
do {
  System.out.print( “I =” + i);
  i++;
  response = readInt( “Enter ” );
}while (i<= 10);
```

Break

- A break statement causes an exit from the innermost containing while, do, for or switch statement.

```
for ( int i = 0; i < maxID, i++ ) {
  if ( userID[i] == targetID ) {
   index = i;
   break;
  }
}// program jumps here after break
```

Continue

- Can only be used with while, do or for.
- The continue statement causes the innermost loop to start the next iteration immediately

```
for ( int i = 0; i < maxID; i++ ) {
  if ( userID[i] != -1 ) continue;
  System.out.print( “UserID ” + i + “ :” +
  userID);
}
```

Review Question

Q.1 Which of the following is not a legal identifier?

Select all valid answers.

(a) a2z

(b) ödipus

(c) 52pickup

(d) _class

(e) ca$h

Answer : 52pickup is not a legal identifier. The first character of an identifier cannot be a digit.

Review Question

Q. 3 Is this a complete and legal comment?

```
/* // */
```

Select the one right answer.

(a) No, the block comment (/* ... */) is not ended since the single-line comment (// ...) comments out the closing part.

(b) It is a completely valid comment. The // part is ignored by the compiler.

(c) This combination of comments is illegal and the compiler will reject it.

Answer : (b) It is a completely valid comment. Comments do not nest. Everything from the start sequence of a multiple-line comment (/*) to the first occurrence of the end sequence of a multiple-line comment (*/) is ignored by the compiler.

Review Question

Q. 4 Which of the following does not denote a primitive data value in Java?

Select all valid answers.

(a) "t"

(b) 'k'

(c) 50.5F

(d) "hello"

(e) false

Answer : (a) and (d) String is a class, and "hello" and "t" denote String objects. Java has the following primitive data types:
boolean, byte, short, char, int, long, float, and double.

Review Question

Q. 5 Which of the following primitive data types are not integer types?

Select the three correct answers.

(a) boolean

(b) byte

(c) float

(d) short

(e) double

Answer : *a), (c), and (e)*

(a) is a boolean data type, while (c) and (e) are floating-point data types.

Review Question

Q. 6 Which of the following lines are valid declarations?

Select all valid answers.

(a) char a = '\u0061';

(b) char 'a' = 'a';

(c) char \u0061 = 'a';

(d) ch\u0061r a = 'a';

(e) ch'a'r a = 'a';

Answer : *(a), (c) and (d)*

Review Question

Q. 7 Which integral type in Java has the exact range from -2147483648 (-2^{31}) to 2147483647 ($2^{31}-1$), inclusive?

Select the one right answer.

(a) Type byte

(b) Type short

(c) Type int

(d) Type long

(e) Type char

Answer : *(c)*

The bit representation of int is 32-bits wide and can hold values in the range –231 through 231–1.

Review Question

Q. 8 Given the following code within a method, which statement is true?

```
int a, b;
b = 5;
```

Select the one correct answer.

(a) Local variable a is not declared.

(b) Local variable b is not declared.

(c) Local variable a is declared but not initialized.

(d) Local variable b is declared but not initialized.

(e) Local variable b is initialized but not declared.

Answer : *(c)*

Local variable a is declared but not initialized. The first line of code declares the local variables a and b. The second line of code initializes the local variable b. Local variable a remains uninitialized.

Review Question

Q. 9 In which of these variable declarations will the variable remain uninitialized unless it is explicitly initialized?

Select the one correct answer.

(a) Declaration of an instance variable of type int.

(b) Declaration of a static variable of type float.

(c) Declaration of a local variable of type float.

(d) Declaration of a static variable of type Object.

(e) Declaration of an instance variable of type int[].

Answer : *(c)*

The local variable of type float will remain uninitialized. Fields and static variables are initialized with a default value. Local variables remain uninitialized unless explicitly initialized. The type of the variable does not affect whether a variable is initialized or not.

.

Review Question

Q. 10 What will be the result of compiling and running the following program?

```
ublic class Init {
    String title;
    boolean published;
    static int total;
    static double maxPrice;
ublic static void main(String[] args) {
    Init initMe = new Init();
    double price;
    if (true)
    price = 100.00;
    System.out.println("|" + initMe.title + "|" + initMe.published + "|" + Init.total + "|" + Init.maxPrice + "|" + price+ "
}
```

Review Question Q. 10...

Select the one correct answer.

(a) The program will fail to compile.

(b) The program will compile, and print |null|false|0|0.0|0.0|, when run.

(c) The program will compile, and print |null|true|0|0.0|100.0|, when run.

(d) The program will compile, and print | |false|0|0.0|0.0|, when run.

(e) The program will compile, and print |null|false|0|0.0|100.0|, when run.

Answer : *(e)*

The program will compile. The compiler can figure out that the local variable price will be initialized, since the value of the condition in the if statement is true. The two instance variables and the two static variables are all initialized to the respective default value of their type.

Review Question

Q. 11

```
class Test{
final static public void main(String args[]){
 System.out.println("Welcome in Java World"); }}
```

(a) Error on compile time.

(b) Compile but error on run time.

(c) Compile and Run with out error. It prints Welcome in Java World.

(d None

Answer : *(c)*

Review Question

Q. 12 What will be the result of attempting to compile and run the following class?

```
public class IfTest {
public static void main(String[] args) {
    if (true)
    if (false)
    System.out.println("a");
    else
    System.out.println("b");
} }
```

Answer : *(d)*
The program will display the letter b when run. The second if statement is evaluated since the boolean expression of the first if statement is true. The else clause belongs to the second if statement. Since the boolean expression of the second if statement is false, the if block is skipped and the else clause is executed.

Select the one correct answer.

(a) The code will fail to compile because the syntax of the if statement is incorrect.

(b) The code will fail to compile because the compiler will not be able to determine which if statement the else clause belongs to.

(c) The code will compile correctly and display the letter a, when run.

(d) The code will compile correctly and display the letter b, when run.

(e) The code will compile correctly, but will not display any output.

Review Question

Q. 13 Which statements are true?

Select the three correct answers.

(a) The conditional expression in an if statement can have method calls.

(b) If a and b are of type boolean, the expression (a = b) can be the conditional expression of an if statement.

(c) An if statement can have either an if clause or an else clause.

(d) The statement if (false) ; else ; is illegal.

(e) Only expressions which evaluate to a boolean value can be used as the condition in an if statement.

Answer : *(a), (b), and (e)*

Review Question

Q. 14 What, if anything, is wrong with the following code?

```
void test(int x) {
switch (x) {
case 1:
case 2:
case 0:
default:
case 4:
} }
```

Select the one correct answer.

(a) The variable x does not have the right type for a switch expression.

(b) The case label 0 must precede case label 1.

(c) Each case section must end with a break statement.

(d) The default label must be the last label in the switch statement.

(e) The body of the switch statement must contain at least one statement.

(f) There is nothing wrong with the code.

Answer : *(f)*

Review Question

Q. 15 Which of these combinations of switch expression types and case label value types are legal within a switch statement?

Select the two correct answers.

(a) switch expression of type int and case label value of type char.

(b) switch expression of type float and case label value of type int.

(c) switch expression of type byte and case label value of type float.

(d) switch expression of type char and case label value of type long.

(e) switch expression of type boolean and case label value of type boolean.

(f) switch expression of type Byte and case label value of type byte.

(g) switch expression of type byte and case label value of type Byte.

Answer : (a) and *(f)*

.

Review Question

Q. 16 Given the following declaration:

```
char c = 'A';
```

What is the simplest way to convert the character value in c into an int?

Select the one correct answer.

(a) int i = c;

(b) int i = (int) c;

(c) int i = Character.getNumericValue(c);

Answer : *(a)*

A value of type char can be assigned to a variable of type int. An widening conversion will convert the value to an int.

Review Question

Q. 17What will be the result of compiling and running the following program?

```
public class Assignment {

public static void main(String[] args) {

int a, b, c;

b = 10;

a = b = c = 20;

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

Select the one correct answer.

Answer : *(d)* An assignment statement is an expression statement. The value of the expression statement is the value of the expression on the right-hand side. Since the assignment operator is right associative, the statement a = b = c = 20 is evaluated as follows:
(a = (b = (c = 20))). This results in the value 20 being assigned to c, then the same value being assigned to b and finally to a. The program will compile, and print 20, when run.

(a) The program will fail to compile since the compiler will report that the variable c in the multiple assignment statement a = b = c = 20; has not been initialized.

(b) The program will fail to compile, because the multiple assignment statement a = b = c = 20; is illegal.

(c) The code will compile and print 10, when run.

(d) The code will compile and print 20, when run.

Review Question

Q.18 What will be the result of compiling and running the following program?

```
public class MyClass {
public static void main(String[] args) {
String a, b, c;
c = new String("mouse");
a = new String("cat");
b = a;
a = new String("dog");
c = b;
System.out.println(c); } }
```

Answer : *(c)*

Strings are objects. The variables a, b, and c are references that can denote such objects. Assigning to a reference only changes the reference value. It does not create a copy of the source object or change the object denoted by the old reference value in the target reference. In other words, assignment to references only affects which object the target reference denotes. The reference value of the "cat" object is first assigned to a, then to b, and later to c. The program prints the string denoted by c, i.e., "cat".

Select the one correct answer.

(a) The program will fail to compile. (b) The program will print **mouse**, when run.

(c) The program will print **cat**, when run. (d) The program will print **dog**, when run.

(e) The program will randomly print either **cat** or **dog**, when run.

Review Question

Q. 19 **Which of the following expressions will be evaluated using floating-point arithmetic?**

Select the three correct answers.

(a) 2.0 * 3.0

(b) 2 * 3

(c) 2/3 + 5/7

(d) 2.4 + 1.6

(e) 0x10 * 1L * 300.0

Answer *(a), (d), and (e)*

A binary expression with any floating-point operand will be evaluated using floating-point arithmetic. Expressions such as 2/3, where both operands are integers, will use integer arithmetic and evaluate to an integer value. In (e), the result of (0x10 * 1L) is promoted to a floating-point value.

Review Question

Q. 20 What is the value of the expression (1 / 2 + 3 / 2 + 0.1)?

Select the one correct answer.

(a) 1

(b) 1.1

(c) 1.6

(d) 2

(e) 2.1

Answer : *(b)*

The / operator has higher precedence than the + operator. This means that the expression is evaluated as ((1/2) + (3/2) + 0.1). The associativity of the binary operators is from left to right, giving (((1/2) + (3/2)) + 0.1). Integer division results in ((0 + 1) + 0.1) which evaluates to 1.1.

Review Question

Q. 21 What will be the result of compiling and running the following program?

```
public class Integers {
public static void main(String[] args) {
System.out.println(0x10 + 10 + 010);
} }
```

Select the one correct answer.

(a) The program will not compile because of errors in the expression 0x10 + 10 + 010.

(b) When run, the program will print 28.

(c) When run, the program will print 30.

(d) When run, the program will print 34.

(e) When run, the program will print 36.

(f) When run, the program will print 101010.

Answer : *(d)* 0x10 is a hexadecimal literal equivalent to the decimal value 16. 10 is a decimal literal. 010 is an octal literal equivalent to the decimal value 8. The println() method will print the sum of these values, which is 34, in decimal form.

Review Question

Q. 22 Which of the following expressions are valid?

Select the three correct answers.

(a) (- 1 -)

(b) (+ + 1)

(c) (+-+-+-1)

(d) (--1)

(e) (1 * * 1)

(f) (- -1)

Answer : *(b), (c), and (f)*

The unary + and - operators with right-to-left associativity are used in the valid expressions (b), (c), and (f). Expression (a) tries to use a nonexistent unary – operator with left-to-right associativity, expression (d) tries to use a decrement operator (--) on an expression that does not resolve to a variable, and expression (e) tries to use a nonexistent unary * operator.

Review Question

Q. 23 What is the value of evaluating the following expression (- -1-3 * 10 / 5-1)?

Select the one correct answer.

(a) –8

(b) –6

(c) 7

(d) 8

(e) 10

(f) None of the above.

Answer : *(b)*

The expression evaluates to –6. The whole expression is evaluated as (((-(-1)) - ((3 * 10) / 5)) - 1) according to the precedence and associativity rules.

Review Question

Q. 24 Which of these assignments are valid?

Select the four correct answers.

(a) short s = 12;

(b) long l = 012;

(c) int other = (int) true;

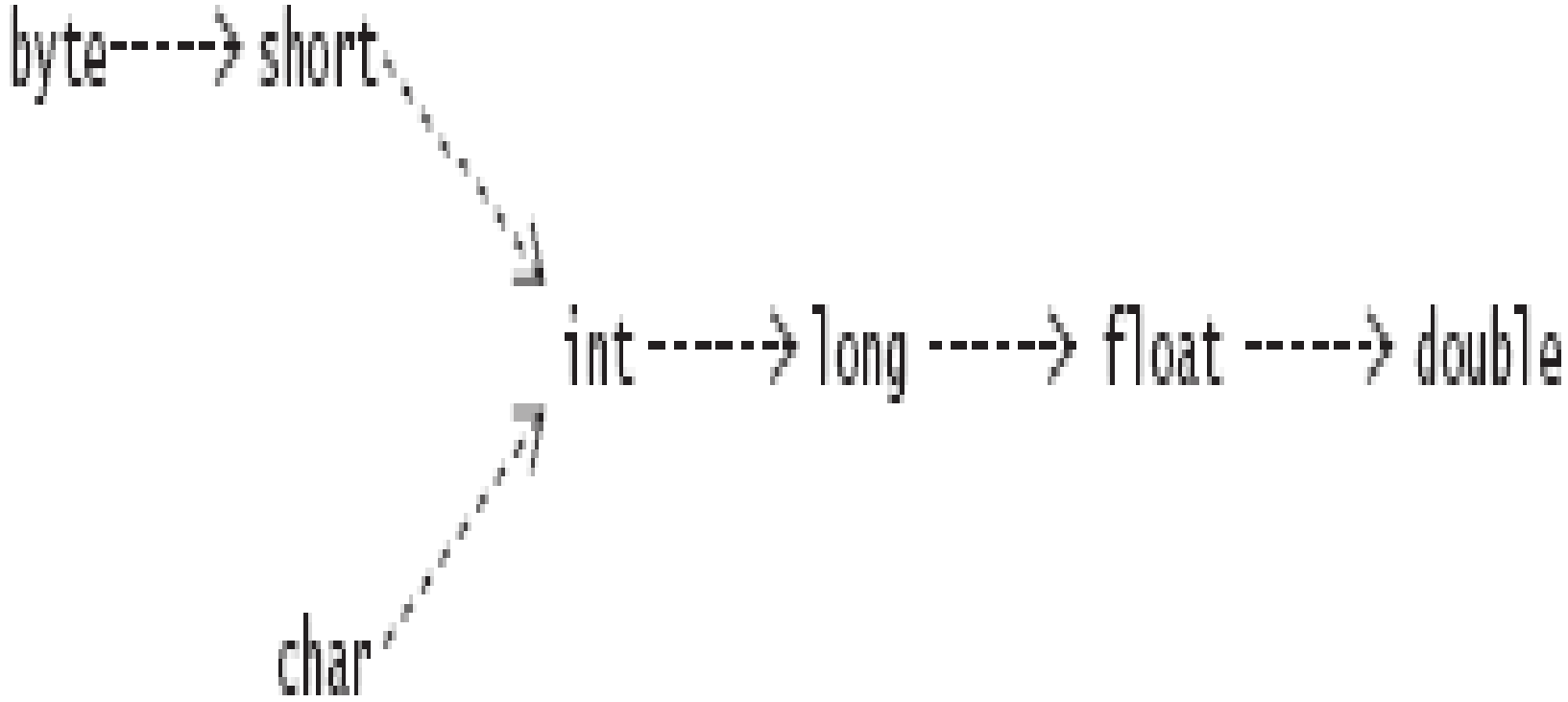
(d) float f = -123;

(e) double d = 0x12345678;

Answer : *(a), (b), (d), and (e)*

In (a), the conditions for implicit narrowing conversion are fulfilled: the source is a constant expression of type int, the destination type is of type short, the value of the source (12) is in the range of the destination type. The assignments in (b), (d), and (e) are valid, since the source type is narrower than the target type and an implicit widening conversion will be applied. The expression (c) is not valid. Values of type boolean cannot be converted to other types.

Widening Primitive Conversions



Review Question

Q. 25 Which statements are true?

Select the three correct answers.

(a) The expression (1 + 2 + "3") evaluates to the string "33".

(b) The expression ("1" + 2 + 3) evaluates to the string "15".

(c) The expression (4 + 1.0f) evaluates to the float value 5.0f.

(d) The expression (10/9) evaluates to the int value 1.

(e) The expression ('a' + 1) evaluates to the char value 'b'.

Answer : *(a), (c), and (d)*

The left associativity of the + operator makes the evaluation of (1 + 2 + “3”) proceed as follows: (1 + 2) + “3” →3 + "3" → "33". Evaluation of the expression ("1“ + 2 + 3), however, will proceed as follows: ("1" + 2) + 3 → "12" + 3 → "123". (4 + 1.0f) evaluates as 4.0f + 1.0f → 5.0f and (10/9) performs integer division, resulting in the value 1. The operand 'a' in the expression ('a' + 1) will be promoted to int, and the resulting value will be of type int.

Review Question

Q. 26 What happens when you try to compile and run the following program?

```
public class Prog1 {
public static void main(String[] args) {
int k = 1;
int i = ++k + k++ + + k; // (1)
System.out.println(i);
} }
```

Answer : *(d)*
The expression ++k + k++ + + k is evaluated as ((++k) + (k++)) + (+k) → ((2) + (2) + (3)), resulting in the value 7.

Select the one correct answer.

(a) The program will not compile, because of errors in the expression at (1).

(b) The program will compile and print the value 3, when run.

(c) The program will compile and print the value 4, when run.

(d) The program will compile and print the value 7, when run.

(e) The program will compile and print the value 8, when run.

Review Question

Q. 27 What is the label of the first line that will cause a compile time error in the following program?

```
public class MyClass {
public static void main(String[ ] args) {
char c; int i;
c = 'a'; // (1)
i = c; // (2)
i++; // (3)
c = i; // (4)
c++; // (5)} }
```

Select the one correct answer.

(a) (1) (b) (2) (c) (3) (d) (4) (e) (5)

Answer : *(d)*

The types char and int are both integral. A char valu
can be assigned to an int variable since the int ty
is wider than the char type and an implicit wideni
conversion will be done. An int type cannot
assigned to a char variable because the char type
narrower than the int type. The compiler will repo
an error about a possible loss of precision in (4).

Review Question

Q. 28 What is the result of compiling and running the following program?

```
public class Cast {
public static void main(String[] args) {
byte b = 128;
int i = b;
System.out.println(i);
} }
```

Answer : *(c)* Variables of the type byte can store values in the range –128 to 127. The expression on the right-hand side of the first assignment is the int literal 128. Had this literal been in the range of the byte type, an implicit narrowing conversion would have been applied to convert it to a byte value during assignment. Since 128 is outside the range of the type byte, the program will not compile.

Select the one correct answer.

(a) The program will not compile because a byte value cannot be assigned to an int variable without using a cast.

(b) The program will compile and print 128, when run.

(c) The program will not compile because the value 128 is not in the range of values for the byte type.

(d) The program will compile but will throw a ClassCastException when run.

(e) The program will compile and print 255, when run.

Review Question

Q. 29 What will be the result of compiling and running the following program?

```
public class EvaluationOrder {
public static void main(String[] args) {
int[] array = { 4, 8, 16 };
int i=1;
array[++i] = --i;
System.out.println(array[0] + array[1] + array[2]);
} }
```

Select the one correct answer.

(a) 13 (b) 14 (c) 20 (d) 21 (e) 24

Answer : *(a)*

First, the expression ++i is evaluated, resulting in the value 2. Now the variable i also has the value 2. The target of the assignment is now determined to be the element array[2]. Evaluation of the right-hand expression, --i, results in the value 1. The variable i now has the value 1. The value of the right-hand expression 1 is then assigned to the array element array[2], resulting in the array contents to become {4, 8, 1}. The program sums these values and prints 13.

Assignment

1. Explain various features of java.
2. What is a JVM? Justify java is plateform neutral language.
3. Explain various operator in Java. What is operator precedence and associtivity?
4. Describe various data type and literal in java.
5. What do mean by garbage collection? How it is supported by Java?
6. Explain the java program sturcutre.
7. Wirte short notes on following java technologies:
   a) Beans
   b) RMI
   c) Servlets
   d) JSP
   e) JSF
   f) CORBA