Principles of Computer Science I

Prof. Nadeem Abdul Hamid

CSC 120A - Fall 2004

Lecture Unit 2

Binary numbers

- See the lecture handout
  - *cs120-04fall-lec01-binarynums.doc*

Review Chapter 1 Goals

- To understand what a computer program is
- To know the three phases of the software life cycle
- To understand what an algorithm is
- To learn what a high-level programming language is
- To understand the difference between machine code and bytecode
- To understand compilation, execution, and interpretation
- To learn the major components of a computer and how they work together
  - ALU, control unit, memory, input, output, auxiliary storage
  - ALU + control = CPU (central processing unit)

Chapter 1 Review, *cont.*

Be able to…

- Distinguish between hardware and software
- List the ways of structuring code in a Java application
  - sequence, selection, loop, subprogram
- Name several problem-solving techniques
- Identify the objects in a problem statement

Writing Java Programs

- Writing programs in any language requires understanding the syntax and semantics of the programming language as well as language-independent skills in programming
- Syntax is similar to rules of spelling and grammar:
  - *i* before *e* except after *c*
- Semantics is what a program (or English sentence) means
- Natural languages are more forgiving than programming languages

Syntax

- Formal set of rules that defining exactly what combinations of letters, numbers, and symbols can be used
- Can describe syntax of a programming language using *another* language called a *meta-language*
  - Backus-Naur form (BNF)
  - Syntax diagrams
  - Syntax templates

Java Identifiers (Names) in BNF

<identifier> ::= <letter> | <letter> <ltr-dgt-seq>

<ltr-dgt-seq> ::= <ltr-dgt> | <ltr-dgt> <ltr-dgt-seq>

<ltr-dgt> ::= <letter> | <digit>

<letter> ::= _ | $ | A | B | ... | Z | a | b | ... | z

<digit> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

Java Identifiers

- Names you give to things (classes, objects, variables, methods, files) in Java
- At least one letter, underscore, or dollar sign
- May be followed by any number of letters, digits, underscores, or dollar signs
- Uppercase and lowercase letters are different
- Use meaningful identifiers in your programs
  - Helps human readers understand your code
- Valid:
  sum_of_squares J9 box_22A GetData
  Bin3D4 count Count
- Invalid:
  40Hours Get Data box-22 empty_? int

Java Reserved Words

- Cannot be used as identifiers

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

Writing a Java Program

- Define a class with a `main` method
- When you run a Java program, it starts from the `main` method
- The primary function of a Java class is to serve as a pattern for objects
- A Java object is a collection of data values and associated operations
- A Java program involves the interaction of objects and classes with each other

Comments

- Ignored by the computer (compiler)
- Comments make programs easier to understand for humans
- Use comments liberally, but make them meaningful
- Two forms of Java comments
  - Comments between `/*` and `*/` can extend over several lines
  - Using two slashes `//` makes the rest of the line become a comment

Methods

- A Java method is a subprogram: a collection of operations that you can call (invoke) with parameters
- The main method is called (invoked) when you first run your program
- Methods can reserve space to store data in local variables
- Each local variable has a name and a type

```
char someLetter;
```

Assignment and Expressions

- We set or change the value of a variable using an assignment statement

```
someLetter = 'A';
```

  - Only one variable can appear to the left of =
  - To the right of = is an expression
  - Value of the expression must have the same type as the variable
- An expression is an arrangement of identifiers, literals, and operators that computes to a value of a given type
- To evaluate means to compute a value by performing a set of operations on given values

Data Types: char

- `char` describes data consisting of one alphanumeric character – letter, digit, or special symbol
  - To write a `char`, use single quotes:
    'A' 'a' '8' '2' '+' '%' '?' ' '
  - Use escape sequences for single quote and backslash:
    '\'' '\\'
- Each `char` is stored at an address in memory
  - Reference the address using a variable identifier
- char is a primitive Java data type

Example program: char

```
public class CharLecture {
    public static void main(String args[]) {
        char someCharacter;
        char someLetter = 'Y';
        someCharacter = '?';

        System.out.print(someLetter);
        System.out.println(someCharacter);
    }
}
```


Data Types: String

- A string is sequence of characters, enclosed in double quotes
- The `String` class is a Java Object type
- Strings in Java are objects (instances of the `String` class)
  "This is " "a String"
- Must be typed all on one line between " "
- Can use escape sequences:
  "She said, \"Hi!.\""
- Example program: *PrintAName.java*

Assignment: Primitive Types vs. Objects

- For a primitive type, the value is stored in the memory location corresponding to the variable identifier
  - Assigning primitive type variables to each other means copying the value in memory
- For objects, a variable contains a value that is the memory address where the actual object is stored
  - Java automatically finds the object in memory using the address (we don't worry about it)
  - But, assigning object variables to each other means copying addresses, not the entire object

Method definitions

- Methods are defined by a heading and a block (of statements)
- Heading tells:
  - Modifiers (access, etc.)
  - Return type
  - Name
  - Parameters
- Block is a sequence of statements enclosed in a pair of braces: { <statements> }
  - For code readability, we indent the body of a block

Method example

```
public class OneMethod {
  public static void printName(String first, char midInit,
                                                String last) {
    System.out.println(first + " " + midInit + ". " + last);
  }

  public static void main(String args[]) {
    printName("Jane", 'S', "Doe");
    printName("John", 'M', "Deer");
  }
}
```


Calling (Invoking) a Method

- Use the name of the method
  - If the method is defined in the same class, just use the name
  - If the method is defined in another class, need to use the name of the class or an object, followed by a . (dot)
  - Example: `System.out.println("Hello");`
- Pass appropriate arguments, enclosed in parentheses, matching the parameter types

Value-returning Methods

- Methods can return a result value

```
public int addTwo(int a, int b) {
  return a + b;
}

…

int result = addTwo(5,19);
```


Fields

- Components of a class that store the data of an object
- May be variables or constants
- A field variable is also called an instance variable, to distinguish from a local variable of a method
- A (named or symbolic) constant is a location in memory, referenced by an identifier, containing a data value that cannot be changed
  - Use the `final` keyword:

```
final char BLANK = ' ';
```


Input/Output

- Output is easy using `System.out.print()` or `println()`
- Input is more complicated, for now use:

```
import java.io.*;        // at the top of the file
…
BufferedReader in = new BufferedReader(new
   InputStreamReader(System.in));
…
in.readLine();        // to read and return a String
```

- Methods using this must also "throw" an Exception (unusual condition or error), again for now

Example

- *Name.java* and *NameDriver.java*

Recap: Types of Methods

- Class methods
  - Associated with a class instead of object instances
  - Defined with the `static` modifier
- Instance methods
  - Methods that operate on the data of an object
- Constructors
  - Special methods called automatically when you create a new object of a class
  - Name must be exactly the same as the class
  - Do not specify a return type (like `void`)
- Void methods
- Value-returning methods
- Helper methods
  - Declared privately in a class; used internally

Recap: Java Applications

- A class or classes containing fields and methods
- Fields: identifier and type
  - Can be variables or constants
- Methods: declarations, statements, expressions, method calls, input, output
- Comments
- One class contains the `main` method