Values, Variables, Types & Arithmetic Expressions

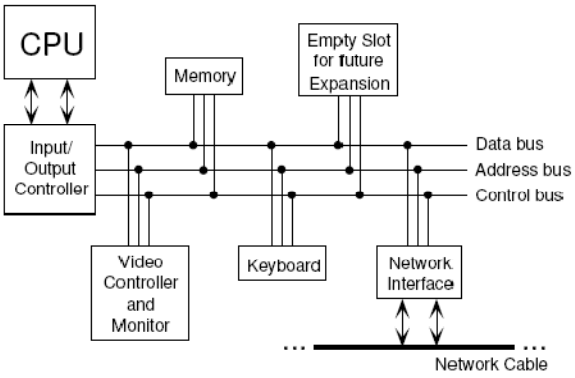
Lecture 2

Object-Oriented Programming

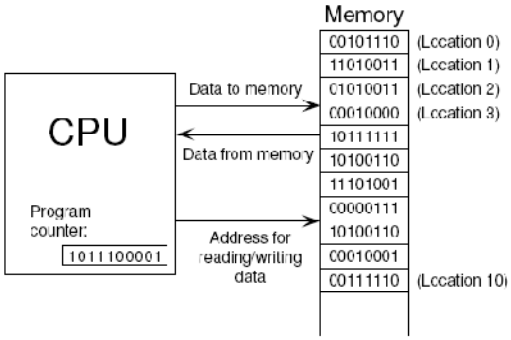
Agenda

- **Inside of a Computer**
- **Value**
- **Variable**
- **Data Types in Java**
- **Literals**
- **Identifiers**
- **Type conversions**
- **Manipulating Variables**
- **Constants**
- **Reserved Words**
- **Manipulating Values**
- **Expression**
- **Operator Precedence and Associativity**
- **Type Conversions**
- **Objects**
- **Readings**

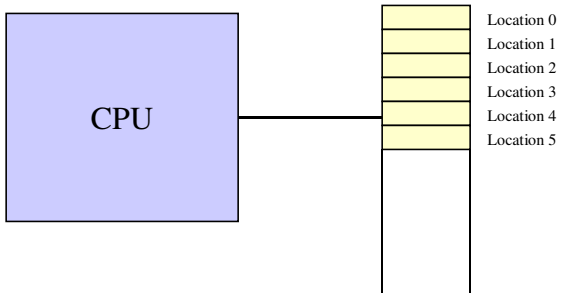
Inside of a Computer



Simplistic View of a Computer



Very Simplistic View of a Computer



This simplistic view is enough to explain the basic concepts of programming to students

Value

- The only task a computer can do is arithmetic e.g. multiplying, dividing, subtracting, etc.
- Therefore, everything in the computer is represented as a value
  - Numbers, letters, characters, etc are all represented as values
- Values could change depending on their nature. For example
  - the temperature today is different from the temperature yesterday
  - The number of cars inside Lahore is different then the number of cars Islamabad.

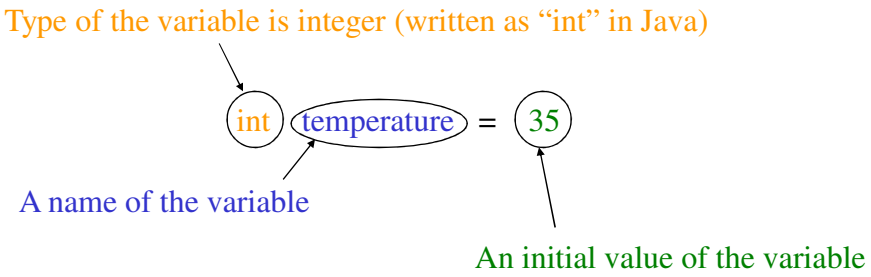
Variable

- To store a value inside a computer a ‘variable’ is used.
- A variable is a space in the memory to store a value.
- This space is reserved until the variable is required.

What Makes a Variable

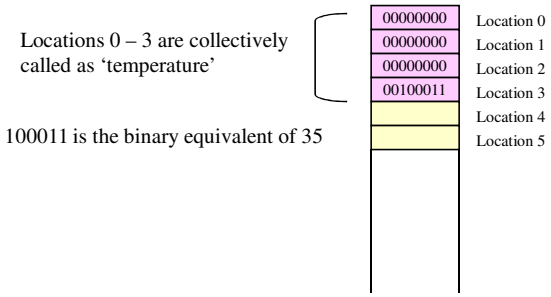
- Variable has three important characteristics:
  - Type
    - How much memory do a variable need.
      - This information is determined by a type.
  - Name
    - How to differentiate a variable with another variable of the same type.
      - Name refers to the memory location assigned to this variable.
  - Value
    - What is the value?
      - The actual value contained by a variable.

An Example of a Variable



Example of a Variable (Memory View)

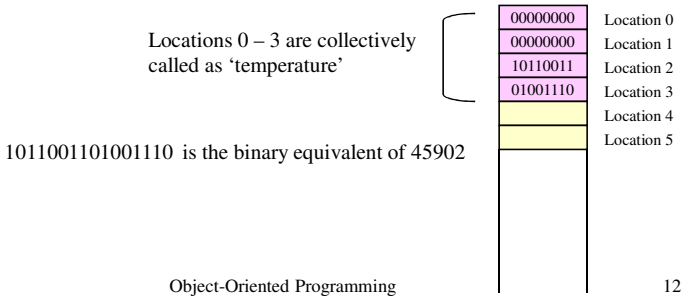
int temperature = 35



Changing the Value of Variable

- Lets change the value of 'temperature'.

temperature = 45902



Type of a Variable

- Among other advantages a ‘type’ binds the memory to a variable name.
- The type int is of 4 bytes in Java.
- Therefore, it can hold maximum of 2,147,483,647 value.
- It can also hold values in negative down to -2,147,483,648.

Variable for Real Numbers

- int cannot hold a real value.
- Therefore, a type “double” is used to hold real values.
- Double takes 8 bytes of memory instead of 4 bytes of a double.
- Out of the 8 bytes in a double 4 bytes are used to hold the value before the decimal point and 4 bytes for the value after the decimal point.

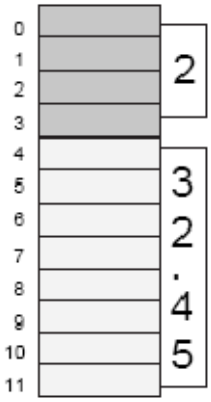
Relative Comparison of int and double

int numPeople = 2;

Reserves 32 bits (4 bytes) and sets the value stored in that space to 2. The name 'numPeople' is associated with this space.

double bill = 32.45;

Reserves 64 bits (8 bytes) and sets the value stored in that space to 32.45. The name 'bill' is associated with this space.



Other Types in Java

Name	Meaning	Range	Size
byte	byte	$-128 \ldots +127$	8 bits
short	short integer	$-32,768 \ldots +32,767$	16 bits
char	Unicode character	$0 \ldots +65,536$	16 bits
int	integer	$-2,147,483,648 \ldots +2,147,483,647$	32 bits
long	long integer	$-9,223,372,036,854,775,808$ $\ldots +9,223,372,036,854,775,807$	64 bits
float	single-precision floating point	$\pm 3.4 \times 10^{+38} \ldots \pm 1.4 \times 10^{-45}$ with at least 7 decimal digits of precision	32 bits
double	double-precision floating point	$\pm 1.7 \times 10^{+308} \ldots \pm 4.9 \times 10^{-324}$ with at least 15 decimal digits of precision	64 bits
boolean	Boolean	false or true	8 bits

Identifier

- Used to denote names of variables, methods and classes.
- Sequence of characters (digits, letters, _, $)
- can't start with a digit.
- Case sensitive
- Select legal identifiers.
  - Sum_, 48my, $$_10, all/clear, get-lost, my48

Literals

- Integer literals
  - 2000, 0, -7 (int)
  - 3000, 3000L (long)
  - Decimal numbers (8)
  - Octal (020)
  - Hexadecimal (0x90)
- Floating point literals
  - 9.8f, 9.8F (float)
  - 56.67, 56.67d (double)

Literals (continued)

- Boolean literals
  - true / false
- Character literals
  - ‘A’, ‘\u0041’
- String literals
  - “Hello”

Type Conversion

- Java can perform conversion automatically
- int value can be assigned to long.
- Depends upon type compatibility
- Not all type conversions implicitly allowed.
- Cant assign a long value to int.
- Solution
  - Casting

Type Conversion (cont.)

- Widening conversion
  - Narrow data types are converted into broad data type with out loss of information
    - Both types are compatible.
      - Numeric types are not compatible with Boolean and char
    - Destination type is larger than source type.
  - Example
    - byte → int
    - int → long

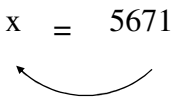
Type Conversion (cont.)

- Narrowing conversion
  - Broader data type is converted into narrower data type with loss of information
  - Process is called casting (explicit type conversion)
  - Target variable = (Target-type) Source variable
    - byte b;
    - int a=50;
    - b=(byte)a;
  - Truncation??????
  - Type conversion in expressions
    - (f*b) + (i/c) –(d*s) ????????

Manipulating Variables

- Assignment Statement
  - In Mathematics the value x = x + 1 is not possible why?
  - In Java x = x +1 is possible because "=" is an assignment operator and not an equality operator.
  - Assignment operator means that the contents of the right hand side is transferred to the memory location of the left hand side.

Assignment Statement



5671 is written at the memory location reserved for x

Constants

- Constants are values which cannot be modified e.g. the value of Pi
- To declare a constant in Java, we write a keyword “final” before the variable type.

final double pi = 3.14;

Reserved Words

- Some names cannot be declared as variable names because they are reserved words in Java

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

Manipulating Values

- Mathematical Operators
  - Common mathematical operators are available in Java for manipulating values e.g. addition(+), subtraction(-), multiplication(*), division(/), and modulus (%).
- Java has many other operators also which we will study in due course.

What is the Result of this Expression?

- What is the result of this arithmetic expression

6 + 2 * 3 / 6

a) 7
b) 0.5
c) 13.0
d) 4

Arithmetic Expression Evaluation

- To evaluate an arithmetic expression two concepts needs to be understood
  - Operator Precedence
    - Operator precedence controls the order in which operations are performed
  - Operator Associativity
    - The associativity of an operator specifies the order in which operations of the same precedence are performed

Operator Precedence and Associativity

- Operators Precedence and Associativity for Java is following

1. *, /, % → Do all multiplications, divisions and remainders from left to right.
2. +, - → Do additions and subtractions from left to right.

Evaluating an Expression

6 + 2 * 3 / 6

- Three operators are in this expression.
- However, * and / both have the same precedence and + has lower precedence then these two.
- * and / will be evaluated first but both have the same precedence level.
- Therefore, operator associativity will be used here to determine the first to get evaluated i.e. left to right.
- The right most sub expression will be evaluated followed by the next right one and so on.

Primitive Data Types

- So far the variable types that we have studied are ***primitive data types***.
- Primitive data types only have a memory space for storing values.
- However, Object-Oriented Programming is special because OOP has more variables then just primitive data types.

Objects

- Objects are one step ahead of primitive data types.
- They contain values and operations both.
  - e.g. A String object has a value as well as operations that could be performed on that value e.g. operations to find its size, operation to compare its size with another String etc.

String Object

- In Java we could use string data type.
- A String in Java is declared as:

  String city = "Lahore";

Thought Question

Why the "S" in String is capitalized?

Readings

Book Name: Object-oriented Programming in JavaTM Textbook

Author: Richard L. Halterman

Content: Chapter 2 and 3

Acknowledgements

- While preparing this course I have greatly benefited from the material developed by the following people:
  - Andy Van Dam (Brown University)
  - Mark Sheldon (Wellesley College)
  - Robert Sedgewick and Kevin Wayne (Princeton University)
  - Mark Guzdial and Barbara Ericsson (Georgia Tech)
  - Richard Halterman (Southern Adventist University)