User Input & Storage

- If a program is receiving input, the information has to be stored somewhere;
 we need a container for the information.
- Information (data) is stored in variables.
- Variables are named memory locations that can vary in value throughout the program.
- Therefore we declare a variable for each piece of information in a program.

Syntax for variable declaration (info in square brackets is optional)

```
datatype identifier [=expression];
// you can initialize a variables at declaration
```

Examples

```
int num;
int a = 5;
float answer;
```

Input

Input: information (data) read in by the program.

For user keyboard input, the Scanner can be used. Note that the Scanner is not available before Java SE 1.5.0.

To read an input from standard input (keyboard), a scanner object needs to be created, and before you can do that, you must import the package java.util:

```
import java.util.*;
Scanner sc = new Scanner(System.in);
```

Then different methods can be used depending on the type of input. For example, to read an integer input from the user:

```
int i = sc.nextInt();
```

The nextInt() method read the input from the user and place it into the variable i.

Other methods to read value of different types from the user:

```
//Returns the byte value (-128 to 127) read from the
byte nextByte()
                           keyboard.
                           //Returns the 2-byte integer (-32,768 to 32,767) read
short nextShort()
                           from the keyboard.
                           //Returns the 4-byte integer (-2,147,483,648 to
int nextInt()
                           2,147,483,647) read from the keyboard.
                           //Returns the 8-byte integer read from the keyboard.
long nextLong()
float nextFloat()
                           //Returns the 4-byte float read from the keyboard.
                           //Returns the 8-byte double read from the keyboard.
double nextDouble()
boolean nextBoolean() //Returns the boolean value (either true or false,
                           case insensitive) read from the keyboard.
String nextLine()
                           //Returns the entire line of input read from the
                           keyboard without the Return
```

Input Syntax

```
import java.util.*;
:
Scanner sc = new Scanner(System.in);
identifer = sc.____();
read method eg. nextInt
```

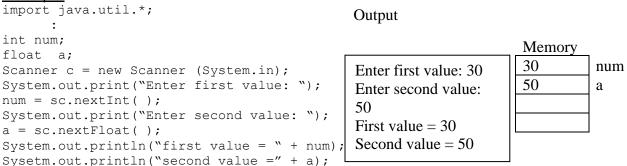
A read statement does the following:

- 1. wait user for input
- 2. store the input from the user to the variable

Outputting variables

To print the values of the variables, use print / println with no quotes around the variable names.

Example



Datatypes

There are many primitive datatypes in any language. These differ in the type of information that can be stored and in the size of the memory location (number of bytes) allocated for the variable.

Keyword	Description	Size	Min Val	Max Val	
(integers3,-2,-1,0,1,2,3)					
byte	Byte-length integer	1 byte	-128	+127	
short	Short integer	2 bytes	-2 ¹⁵	+2 ¹⁵ -1	
int	Integer	4 bytes	-2 ³¹	+2 ³¹ -1	
long	Long integer	8 bytes	-2 ⁶³	+2 ⁶³ -1	
(real numbers; eg. 5.4, 6.7, 8, -200.678)					
float	Single-precision floating point	4 bytes	32-bit IEEE 754 floating-point numbers		
double	Double-precision floating point	8 bytes	64-bit IEEE 754 floating-point numbers		
(other types)					
char	A single character (eg. 'a', '&', '8', '%'	2 bytes	0	2 ¹⁶ -1	
boolean	A boolean value (true or false)	1 bit	true and false		

Examples of Literal Values and Their Data Types

Literal	Data Type		
178	int		
8864L	long		
37.266	double		
37.266D	double		
87.363F	float		
26.77e3	double		
'c '	char		
true	boolean		
false	boolean		

Notes:

- a series of digits with no decimal point is by default considered an integer.
- You can specify a long integer by putting an 'L' or 'l' after the number. 'L' is preferred as it cannot be confused with the digit 'l'.
- A series of digits with a decimal point is by default considered type double.
- You can specify a float by putting an 'f' or 'F' after the number.
- A literal character value is any single Unicode character between single quote marks.
- The two boolean literals are simply true and false.

Valid Variable Names

There are naming conventions for variable strings

Guidelines

- 1) under 256 characters
- 2) only letters, digits and underscores (_)
- 3) can't start with a digit
- 4) no spaces
- 5) no reserved/keywords