Lecture 3B
Data Types and Variables

Course: Object Oriented Programming (CS F213)

Prof. Anita Agrawal
BITS Pilani-K.K.Birla Goa campus

Variables

- A variable is a named memory location capable of storing data

- We can also store data in simple variables, which represent data only, without any associated methods

Literals

Character Literals

- Represented by enclosing in single quotes
  - Example : char c = ‘A’;

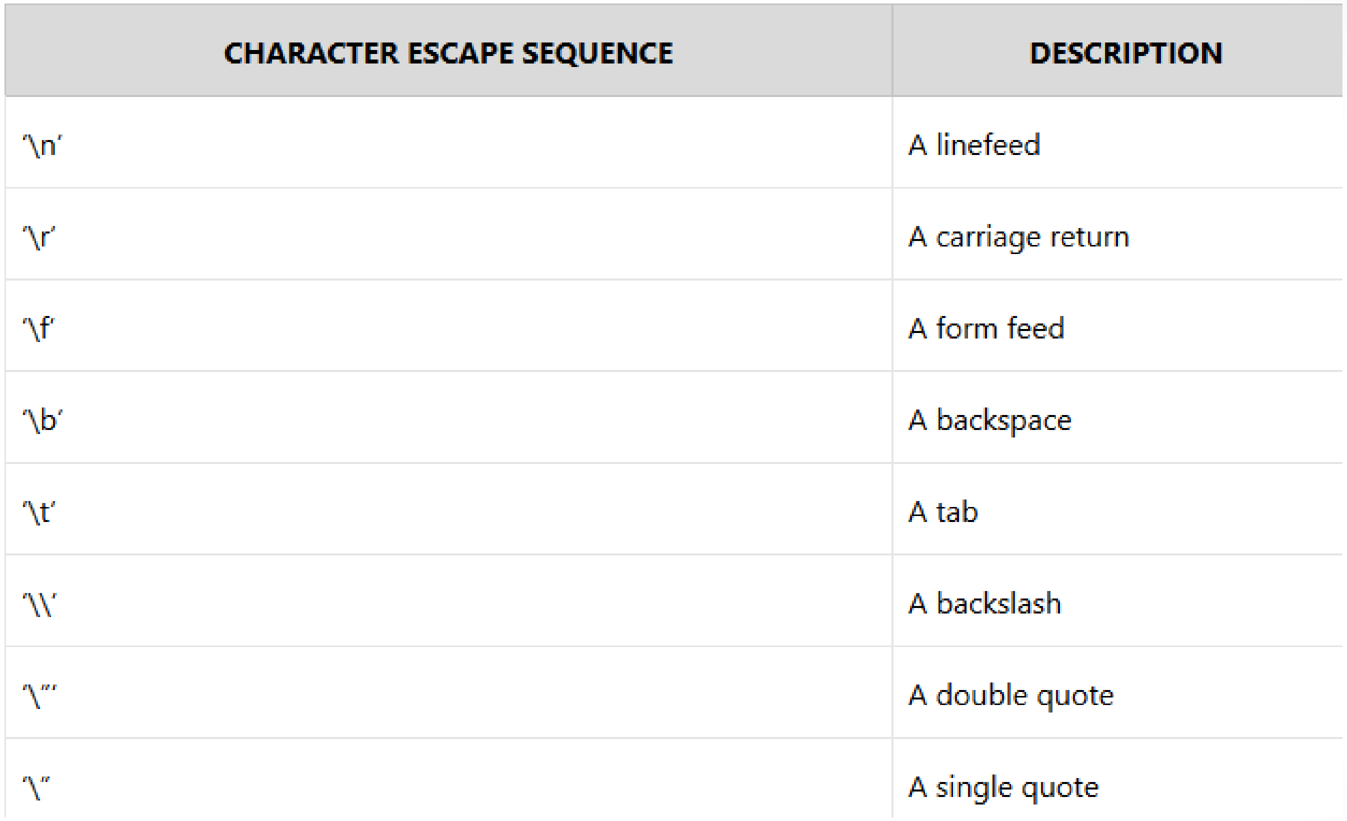
- 16 bit values can be converted into integers and manipulated with integer operators, by enclosing in ‘ ‘.
  - Example : char c = ‘A’;

    ```
    c++;        // c now contains ‘B’
    ```
  - Example:

    char letter = '\u0051';

  - Visible ASCII characters: ‘A’ ‘k’ ‘6’ ‘@’

Escape Sequences in Java



CHARACTER ESCAPE SEQUENCE	DESCRIPTION
‘\n’	A linefeed
‘\r’	A carriage return
‘\f’	A form feed
‘\b’	A backspace
‘\t’	A tab
‘\\’	A backslash
‘\”’	A double quote
‘\’’	A single quote

There are only 8 escape sequences in Java. You cannot define your own character escape sequences

String type

- In addition to the 8 primitive data types, the Java programming language provides special support for character strings via the java.lang.String class.
- It is technically not a primitive data type.
- String objects are immutable.

String Literals

- Strings are implemented as objects rather than an array of characters
- Sequence of characters enclosed in a pair of double quotes.
  - Example: “Hello World!”
  - Example: “These are \n two lines”
  - Example: “\”This is shown in Quotes\””

- Write down a Java program to print the following:

g

Java8

Example char

```
• class  stExample
{
public static void main(String[] args)
    {
        char myChar = 'g';
        char newLine = '\n';
        String myString = "Java 8";
        System.out.println(myChar);
        System.out.println(newLine);
        System.out.println(myString);
    }
}
```

Example char

```
• class  stExample
{
public static void main(String[] args)
    {
        System.out.println(‘g’);
        System.out.println(‘\n’);
        System.out.println(“Java8”);
    }
}
```

Example double: What will be the output of the following code??

```
class DoubleExample
 {
   public static void main(String args [])
        {
            double d =41.2;
            float f = 41.2F;
            double ds = 1.836e3;

        System.out.println(d);
        System.out.println(f);
        System.out.println(ds);
         }
}
```

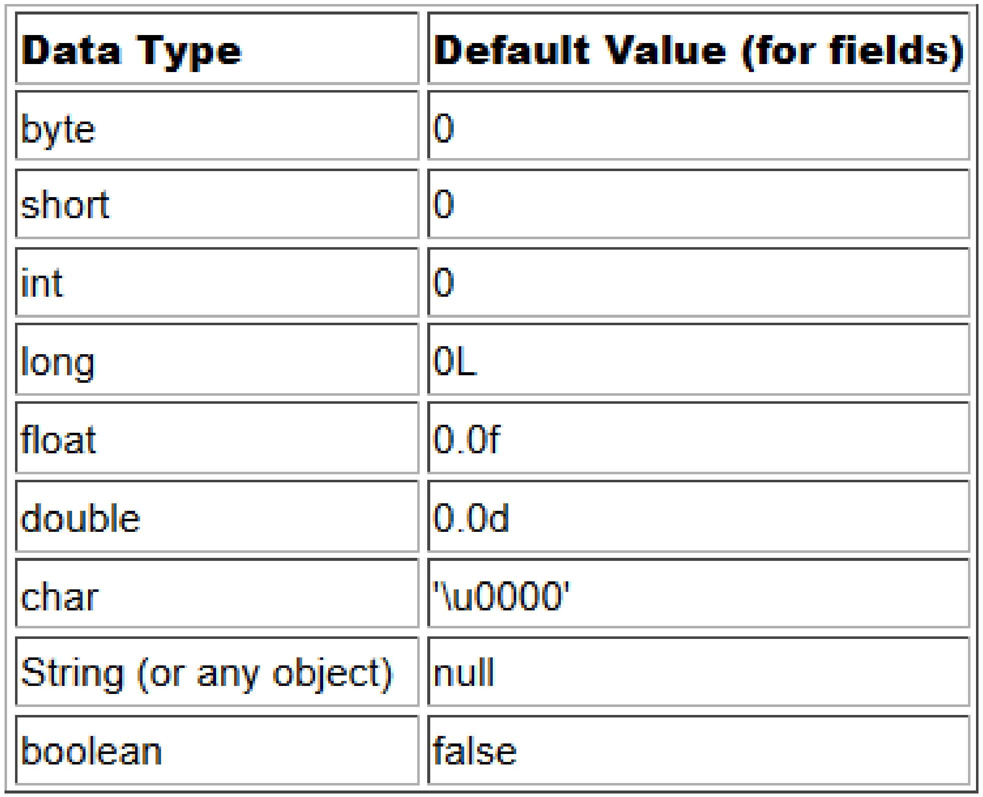
Output

```
41.2
41.2
1836.0
```

Default values

- It's not always necessary to assign a value when a field is declared.
- Fields that are declared but not initialized will be set to a reasonable default by the compiler depending on the data type.
- Relying on such default values, however, is generally considered bad programming style.

Default values of the data types



Data Type	Default Value (for fields)
byte	0
short	0
int	0
long	0L
float	0.0f
double	0.0d
char	'\u0000'
String (or any object)	null
boolean	false

Example: What will be the output of this code?

```
class trial
{
        public static void main(String[] args)
    {
         char c= 'A';
        c= 'B';
         System.out.println(c);
     }
}
```

Output: B

Example: What will be the output of this code?

```
class trial
{
        public static void main(String[] args)
    {
         char c= 'A';
          int i;
         c= 'B';
         System.out.println(c);
           System.out.println(i);


     }
}
```

Output: B????

Example: What will be the output of this code?

```
class trial
{
        public static void main(String[] args)
    {
         String str= “Hi”;
        str= “hello”;
         System.out.println(str);
     }
}
```

Output: hello

Type Conversion

- It is possible to assign the value of one type to the variable of other type, both implicitly (automatically) or explicitly.

- If two types are compatible, automatic conversion is performed.
  - For example, int to long int.

- For incompatible types like char-int or double-byte, this has to be done using type casting.

Automatic Type Conversion

- **Automatic conversion happens when:**
  - **Two types are compatible**
  - **Destination type is larger than the source type**

Automatic type conversion conditions:

- Two types are compatible
  - **Numeric types (integer and floating-point) are compatible with each other**
  - **Numeric types are not automatically converted to char or Boolean**
  - **char and boolean are not compatible with each other**

- Destination type is larger than the source-
- **Widening conversion happens**

Type casting

- This is done when the Destination type is smaller than the source type, i.e. narrowing conversion occurs.

- For example,

```
int i;
byte b;
b = (byte) i; //type casting
i = b;        //type casting not required
```

Example revisited

```
class  autoconvert {
public static void main(String arg[])
{
int i=5;
byte b=10;
short s=15;
long l=20;
i=i+1;
b=b+1;
s=s+1;
i=i+1;
System.out.println("i: "+i+" b: "+b+" s: "+s+" i: "+l);
}
```

Example revisited

```
class  autoconvert {
public static void main(String arg[])
{
int i=5;
byte b=10;
short s=15;
long l=20;
i=i+1;
b=(byte)(b+1); //evaluates to an integer
s=(short)(s+1); //evaluates to an integer
i=i+1;
System.out.println("i: "+i+" b: "+b+" s: "+s+" i: "+l);
}
```

Issues with Explicit Type

- Loss of information.

  If a float is converted to int, the decimal part is lost.

```
int a;
float b =8.923;
a=b;
//a value will be 8
```

If the Target type has smaller range, Value is reduced modulo the target type’s range

```
int i = 257;
        byte b = (byte) i;

//b will store 257%256=1
```

(Byte’s range)

Automatic Type Promotion

- Can also occur in expressions

- Example-

```
byte x = 40;
byte y = 50;
byte z = 100;
int a = (x * z ) + y;
```

- The result of x * z exceeds the range of byte. Java automatically **promotes** byte (and short) to int while evaluating an expression

- It can however cause compile time errors if incompatible types

Rules of Type promotion

- Byte and short are promoted to int
- If one operand is long, the entire expression becomes long
- If one operand is float the entire expression becomes float
- If one expression is double, the entire expression becomes double

Example: Find the output

```
byte b = 42;
char c = 'a';
short s = 1024;
int i = 50000;
float f = 5.67f;
double d = .1234;
double result = (f * b) + (i / c) -(d * s);
```

- 626.778