CS 234

Review – Data types

Data types

- Variables
- Arithmetic
- I/O
- Strings

Variables

- Java is a strongly typed language

```
public class Main{

public static void main(String args[])
{
	int salary;
	System.out.println(salary);
}
}
```

Data types

- Primitive

Type	Size	Range
byte	1 byte	-128 to 127
short	2 bytes	-32,768 to 32,767
int	4 bytes	-2,147,483,648 to 2,147,483,647
long	8 bytes	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
float	4 bytes	±10^38 (~6 decimal digits)
double	8 bytes	±10^308 (~15 decimal digits)
char	2 bytes	characters
boolean	Not really defined (1 bit)	True | False

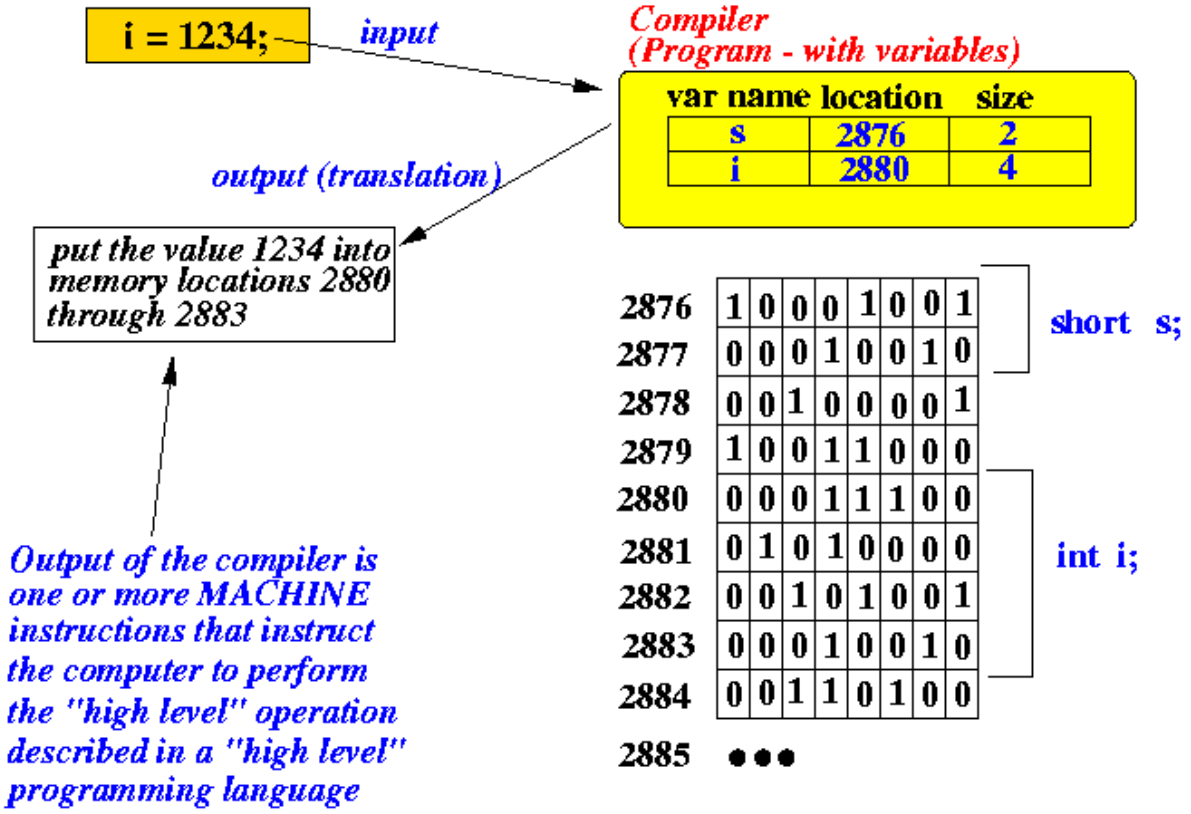
Note:
Strings in Java are objects from the class String

Data types

```
public class Main
{
  public static void main (String[] args)
  {
    System.out.println("Size of byte: " + (Byte.SIZE/8) + " bytes.");
    System.out.println("Size of short: " + (Short.SIZE/8) + " bytes.");
    System.out.println("Size of int: " + (Integer.SIZE/8) + " bytes.");
    System.out.println("Size of long: " + (Long.SIZE/8) + " bytes.");
    System.out.println("Size of char: " + (Character.SIZE/8) + " bytes.");
    System.out.println("Size of float: " + (Float.SIZE/8) + " bytes.");
    System.out.println("Size of double: " + (Double.SIZE/8) + " bytes.");
  }
}
```

```
Size of byte: 1 bytes.
Size of short: 2 bytes.
Size of int: 4 bytes.
Size of long: 8 bytes.
Size of char: 2 bytes.
Size of float: 4 bytes.
Size of double: 8 bytes.
```

Data types



For security, there is no way to access addresses of variables in Java.

Data types

```
class Main {
    public static void main(String[] args)
{

        byte num = 153;
        System.out.println(num);
    }
}
```

What is the output?

Main.java:15: error: incompatible types: possible lossy conversion from int to byte byte num = 153; ^ 1 error

Data types

```
class Main {
    public static void main(String[] args)
{

        float num = 19.98;
        System.out.println(num);
    }
}
```

What is the output?

Main.java:14: error: incompatible types: possible lossy conversion from double to float float num = 19.98; ^ 1 error

Data types

```
class Main {
    public static void main(String[] args) {

        float num = 19.98f;
        System.out.println(num);

System.out.println(((Object)num).getClass().getName());

    }
}
```

What is the output?

19.98

java.lang.Float

Data types

```
class Main {
    public static void main(String[] args) {

        int fiftyMillion = 50_000_000;
         System.out.println(100 * fiftyMillion); // Expected: 5 000 000 000

    }
}
```

What is the output?

705032704

Constants

- Use “final”

```
class Main {
    public static void main(String[] args) {

        final double BOTTLE_VOLUME = .500; // ml

        double totalVolume;
        double bottles = 3;

        totalVolume = bottles * .500; // What does this number represent

        totalVolume = bottles * BOTTLE_VOLUME;

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

Arithmetics

```
class Main {
    public static void main(String[] args) {

        int grade1 = 8;
        int grade2 = 7;
        int grade3 = 8;

        int avg = (grade1 + grade2 + grade3) / 3;

        System.out.println(avg);

    }
}
```

What is the output?

7

```
class Main {
    public static void main(String[] args) {

        int grade1 = 8;
        int grade2 = 7;
        int grade3 = 8;

        double avg = (grade1 + grade2 + grade3) / 3;

        System.out.println(avg);

    }
}
```

What is the output?

7.0

Arithmetics

```
class Main {
    public static void main(String[] args) {

        int grade1 = 8;
        int grade2 = 7;
        int grade3 = 8;

        double avg = (double)(grade1 + grade2 + grade3) / 3;

        System.out.println(avg);

    }
}
```

What is the output?

7.666666666666667

Arithmetics

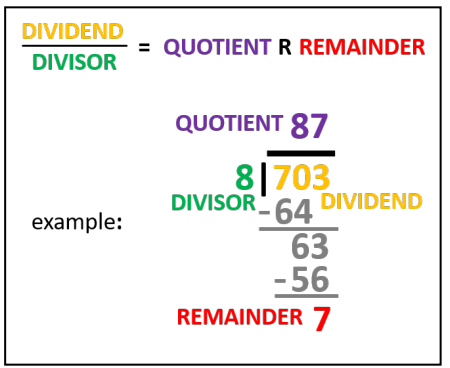
```
class Main {
    public static void main(String[] args) {

        int a = 100;

        System.out.println("a++:"+ a++);
        System.out.println("a--:"+ a--);
        System.out.println("++a:"+ (++a));
        System.out.println("--a:"+ (--a));

    }
}
```

What is the output?

```
a++:100
a--:101
++a:101
--a:100
```

Arithmetics

```java
public class Main
{
    public static void main(String[] args) {

        int number1 = 45;
        int number2 = 246;

        int number = number1;
        if ((number % 2) == 0)
        {
            System.out.println("Quotient: "+ number/2 +" Remainder: " + number%2);
            System.out.println(number + " is even");
        }
        else
        {
            System.out.println("Quotient: "+ number/2 +" Remainder: " + number%2);
            System.out.println(number + " is odd");
        }
    }
}
```




```
Quotient: 22 Remainder: 1
45 is odd
```

Mathematical Methods

Method	Returns
`Math.sqrt(x)`	Square root of x (≥ 0)
`Math.pow(x, y)`	x^y ($x > 0$, or $x = 0$ and $y > 0$, or $x < 0$ and y is an integer)
`Math.sin(x)`	Sine of x (x in radians)
`Math.cos(x)`	Cosine of x
`Math.tan(x)`	Tangent of x
`Math.toRadians(x)`	Convert x degrees to radians (i.e., returns $x \cdot \pi/180$)
`Math.toDegrees(x)`	Convert x radians to degrees (i.e., returns $x \cdot 180/\pi$)
`Math.exp(x)`	e^x
`Math.log(x)`	Natural log ($\ln(x)$, $x > 0$)
`Math.log10(x)`	Decimal log ($\log_{10}(x)$, $x > 0$)
`Math.round(x)`	Closest integer to x (as a `long`)
`Math.abs(x)`	Absolute value $\lvert x \rvert$
`Math.max(x, y)`	The larger of x and y
`Math.min(x, y)`	The smaller of x and y

I/O

- Reading user input

```
import java.util.Scanner;

class Main
{
	public static void main(String args[])
	{
		// Using Scanner for Getting Input from User
		Scanner in = new Scanner(System.in);

		String s = in.nextLine();
		System.out.println("You entered string "+s);

		int a = in.nextInt();
		System.out.println("You entered integer "+a);

		float b = in.nextFloat();
		System.out.println("You entered float "+b);
	}
}
```

```
Hello
You entered string Hello
4
You entered integer 4
4.5
You entered float 4.5
```

I/O

```
Hello
You entered string Hello
4.2
Exception in thread "main" java.util.InputMismatchException
        at java.util.Scanner.throwFor(Scanner.java:864)
        at java.util.Scanner.next(Scanner.java:1485)
        at java.util.Scanner.nextInt(Scanner.java:2117)
        at java.util.Scanner.nextInt(Scanner.java:2076)
        at Main.main(Main.java:24)
```

I/O

Use the ***printf*** statement

Format String	Sample Output	Comments
`"%d"`	`24`	Use `d` with an integer.
`"%5d"`	`  24`	Spaces are added so that the field width is 5.
`"Quantity:%5d"`	`Quantity:  24`	Characters inside a format string but outside a format specifier appear in the output.
`"%f"`	`1.21997`	Use `f` with a floating-point number.
`"%.2f"`	`1.22`	Prints two digits after the decimal point.
`"%7.2f"`	`  1.22`	Spaces are added so that the field width is 7.
`"%s"`	`Hello`	Use `s` with a string.
`"%d %.2f"`	`24 1.22`	You can format multiple values at once.

I/O

```
class Main {
    public static void main(String[] args) {

        int grade1 = 8;
        int grade2 = 7;
        int grade3 = 8;

        double avg = (double)(grade1 + grade2 + grade3) / 3;

        System.out.printf("%.2f",avg);

    }
}
```

7.67

Strings

Statement	Result	Comment
`string str = "Ja";` `str = str + "va";`	`str` is set to `"Java"`	When applied to strings, `+` denotes concatenation.
`System.out.println("Please"` `  + " enter your name: ");`	Prints `Please enter your name:`	Use concatenation to break up strings that don't fit into one line.
`team = 49 + "ers"`	`team` is set to `"49ers"`	Because `"ers"` is a string, 49 is converted to a string.

`String first = in.next();`
`String last = in.next();`
`(User input: Harry Morgan)`
	`first` contains `"Harry"`
	`last` contains `"Morgan"`
		The `next` method places the next word into the string variable.

`String greeting = "H & S";` `int n = greeting.length();`	`n` is set to 5	Each space counts as one character.
`String str = "Sally";` `char ch = str.charAt(1);`	`ch` is set to `'a'`	This is a `char` value, not a `String`. Note that the initial position is 0.
`String str = "Sally";` `String str2 = str.substring(1, 4);`	`str2` is set to `"all"`	Extracts the substring starting at position 1 and ending before position 4.
`String str = "Sally";` `String str2 = str.substring(1);`	`str2` is set to `"ally"`	If you omit the end position, all characters from the position until the end of the string are included.
`String str = "Sally";` `String str2 = str.substring(1, 2);`	`str2` is set to `"a"`	Extracts a `String` of length 1; contrast with `str.charAt(1)`.
`String last = str.substring(` `  str.length() - 1);`	`last` is set to the string containing the last character in `str`	The last character has position `str.length() - 1`.

Do you recall what I said about Strings being objects?

Strings

```
public class Main{
   public static void main(String args[]) {
      String str= new String("Today is a beautiful day to code in Java");
      System.out.println("Original sentence:" + str);
      System.out.println("Substring starting from index 10:");
      System.out.println(str.substring(10));
      System.out.println("Substring starting from index10 and ending at 20:");
      System.out.println(str.substring(10, 20));
   }
}
```

```
Original sentence:Today is a beautiful day to code in Java
Substring starting from index 10:
 beautiful day to code in Java
Substring starting from index 10 and ending at 20:
 beautiful
```

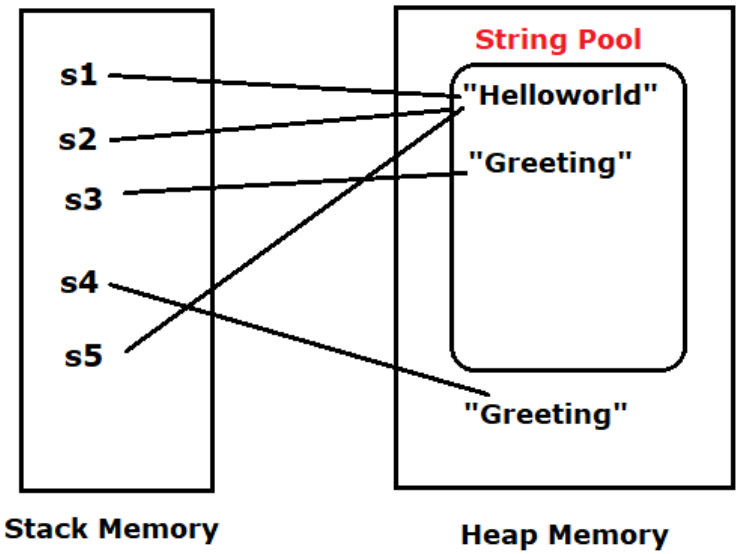
Strings

```
public class Main{
   public static void main(String args[]) {

       String str1 = "Hello";
       String str2 = "Hello";
       System.out.println(str1 == str2);

       String str3 = new String("Hello");
       String str4 = new String("Hello");
       System.out.println(str3 == str4);

       String str5 = "Hello";
       String str6 = "Hello";
       System.out.println(str5.equals(str6));
   }
}
```

```
true
false
true
```

Objects in Java live in the Heap

```
ing s1 = "Helloworld";
ing s2 = "Helloworld";
ing s3 = "Greeting";
ing s4 = new String("Greeting");
ing s5 = "Helloworld";
```



When we create a *String* via the *new* operator, the Java compiler will create a new object and store it in the heap space reserved for the JVM.