# COMPSCI 201 LAB 1

# TA Introduction

- Name: Sara Kojic
- Class of 2026
- Email:
- sara.kojic@duke.edu
- sara.kojic@dukekunshan.edu.cn
- sk794@duke.edu

# IDE

Eclipse

IntelliJ IDEA

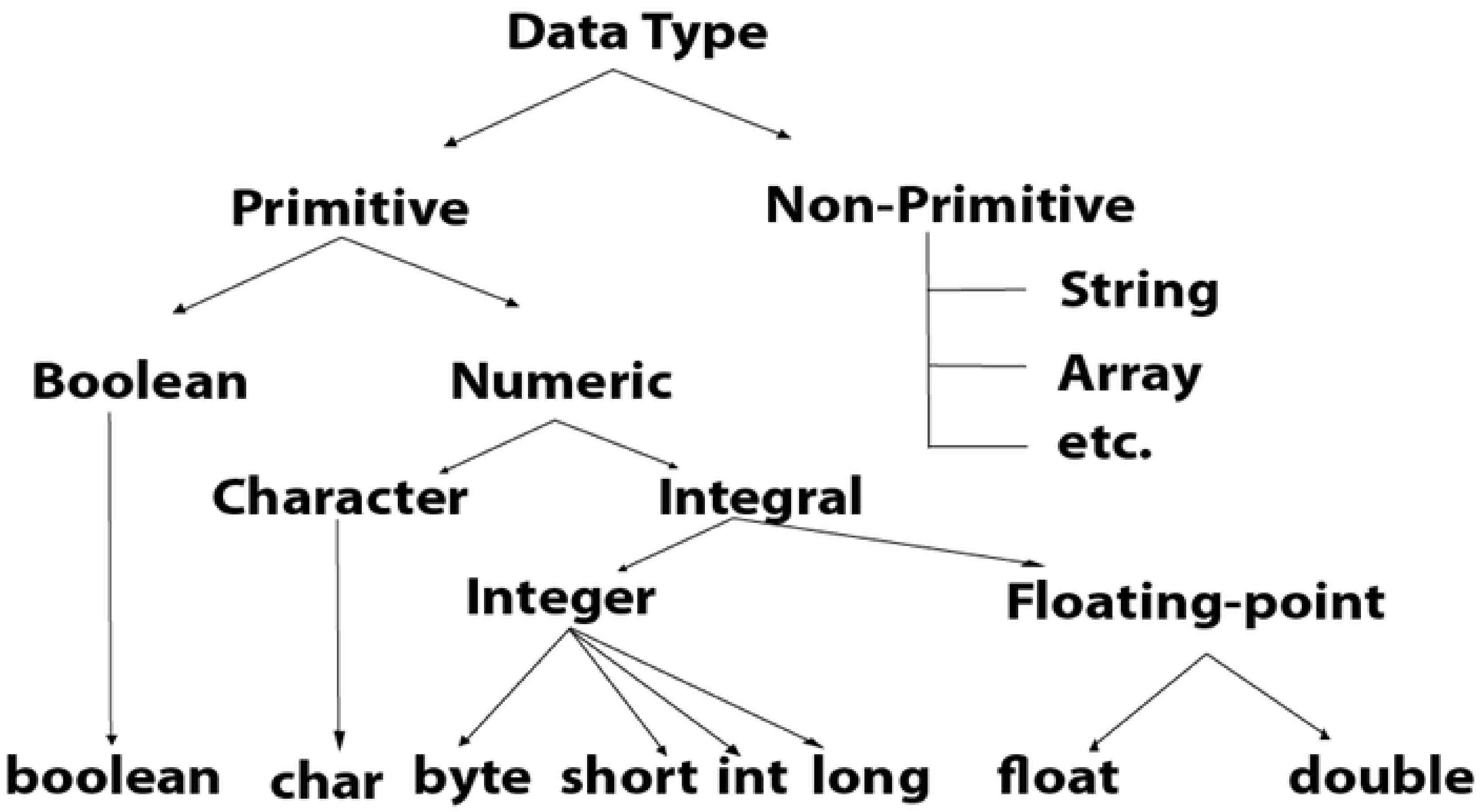
Netbeans

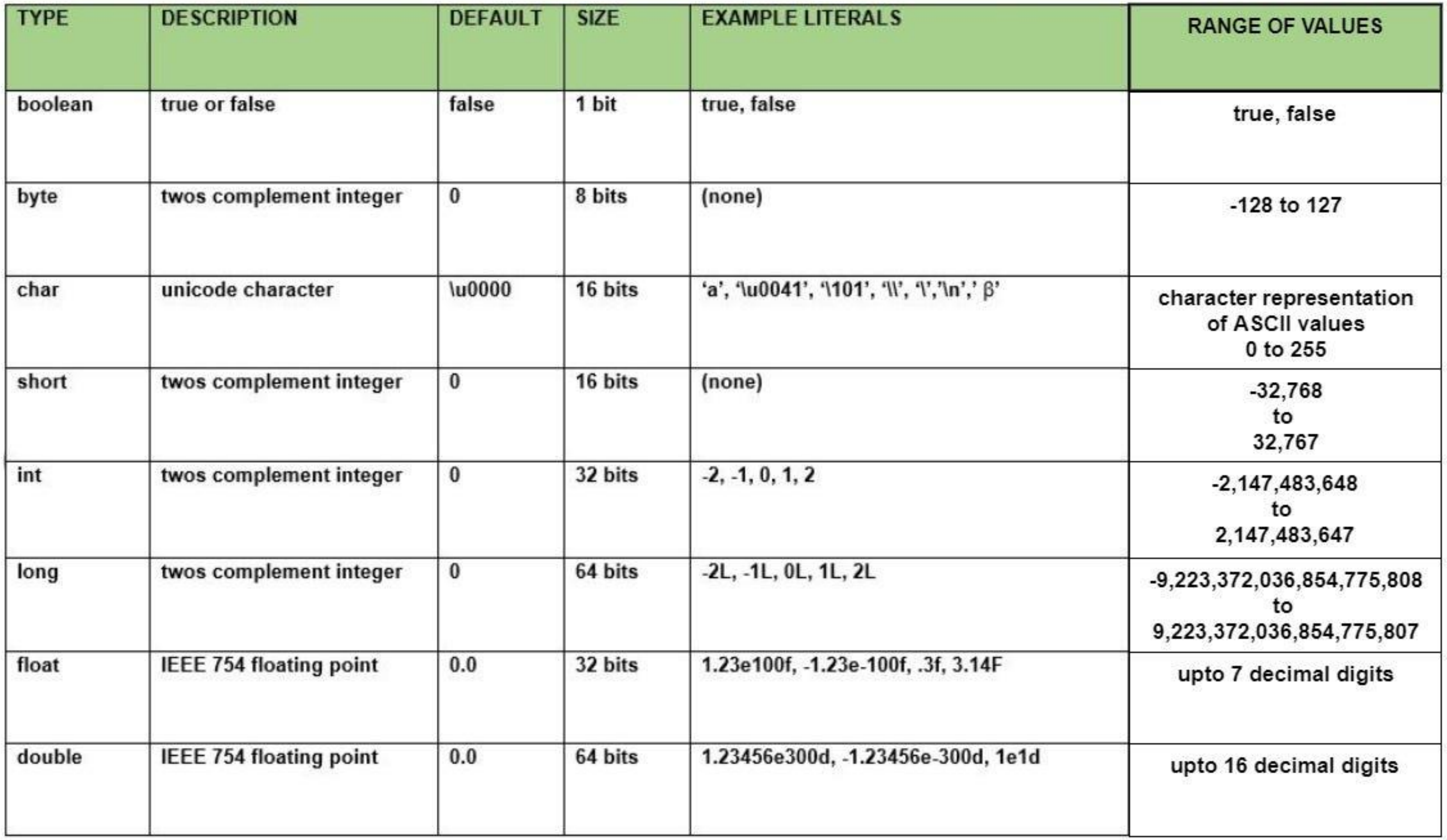
Visual Studio

XCode

Apache ANT

# Data types in Java





| TYPE | DESCRIPTION | DEFAULT | SIZE | EXAMPLE LITERALS | RANGE OF VALUES |
|---|---|---|---|---|---|
| boolean | true or false | false | 1 bit | true, false | true, false |
| byte | twos complement integer | 0 | 8 bits | (none) | -128 to 127 |
| char | unicode character | \u0000 | 16 bits | ‘a’, ‘\u0041’, ‘\101’, ‘\\’, ‘\’,’\n’,’ β’ | character representation<br>of ASCII values<br>0 to 255 |
| short | twos complement integer | 0 | 16 bits | (none) | -32,768<br>to<br>32,767 |
| int | twos complement integer | 0 | 32 bits | -2, -1, 0, 1, 2 | -2,147,483,648<br>to<br>2,147,483,647 |
| long | twos complement integer | 0 | 64 bits | -2L, -1L, 0L, 1L, 2L | -9,223,372,036,854,775,808<br>to<br>9,223,372,036,854,775,807 |
| float | IEEE 754 floating point | 0.0 | 32 bits | 1.23e100f, -1.23e-100f, .3f, 3.14F | upto 7 decimal digits |
| double | IEEE 754 floating point | 0.0 | 64 bits | 1.23456e300d, -1.23456e-300d, 1e1d | upto 16 decimal digits |

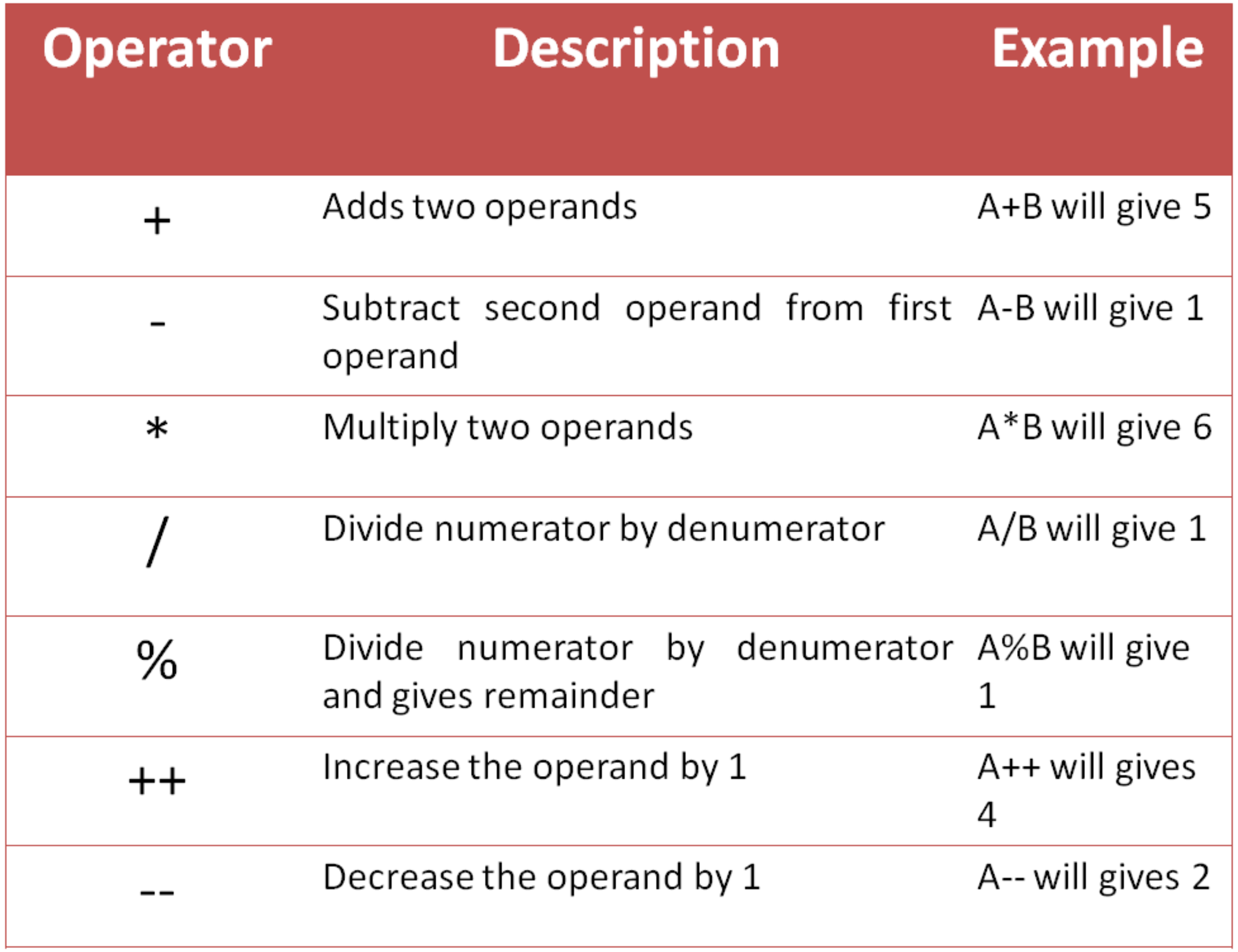
# Decimal - Binary - Octal - Hex – ASCII
# Conversion Chart

| Decimal | Binary | Octal | Hex | ASCII | Decimal | Binary | Octal | Hex | ASCII | Decimal | Binary | Octal | Hex | ASCII | Decimal | Binary | Octal | Hex | ASCII |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 00000000 | 000 | 00 | NUL | 32 | 00100000 | 040 | 20 | SP | 64 | 01000000 | 100 | 40 | @ | 96 | 01100000 | 140 | 60 | ` |
| 1 | 00000001 | 001 | 01 | SOH | 33 | 00100001 | 041 | 21 | ! | 65 | 01000001 | 101 | 41 | A | 97 | 01100001 | 141 | 61 | a |
| 2 | 00000010 | 002 | 02 | STX | 34 | 00100010 | 042 | 22 | " | 66 | 01000010 | 102 | 42 | B | 98 | 01100010 | 142 | 62 | b |
| 3 | 00000011 | 003 | 03 | ETX | 35 | 00100011 | 043 | 23 | # | 67 | 01000011 | 103 | 43 | C | 99 | 01100011 | 143 | 63 | c |
| 4 | 00000100 | 004 | 04 | EOT | 36 | 00100100 | 044 | 24 | $ | 68 | 01000100 | 104 | 44 | D | 100 | 01100100 | 144 | 64 | d |
| 5 | 00000101 | 005 | 05 | ENQ | 37 | 00100101 | 045 | 25 | % | 69 | 01000101 | 105 | 45 | E | 101 | 01100101 | 145 | 65 | e |
| 6 | 00000110 | 006 | 06 | ACK | 38 | 00100110 | 046 | 26 | & | 70 | 01000110 | 106 | 46 | F | 102 | 01100110 | 146 | 66 | f |
| 7 | 00000111 | 007 | 07 | BEL | 39 | 00100111 | 047 | 27 | ' | 71 | 01000111 | 107 | 47 | G | 103 | 01100111 | 147 | 67 | g |
| 8 | 00001000 | 010 | 08 | BS | 40 | 00101000 | 050 | 28 | ( | 72 | 01001000 | 110 | 48 | H | 104 | 01101000 | 150 | 68 | h |
| 9 | 00001001 | 011 | 09 | HT | 41 | 00101001 | 051 | 29 | ) | 73 | 01001001 | 111 | 49 | I | 105 | 01101001 | 151 | 69 | i |
| 10 | 00001010 | 012 | 0A | LF | 42 | 00101010 | 052 | 2A | * | 74 | 01001010 | 112 | 4A | J | 106 | 01101010 | 152 | 6A | j |
| 11 | 00001011 | 013 | 0B | VT | 43 | 00101011 | 053 | 2B | + | 75 | 01001011 | 113 | 4B | K | 107 | 01101011 | 153 | 6B | k |
| 12 | 00001100 | 014 | 0C | FF | 44 | 00101100 | 054 | 2C | , | 76 | 01001100 | 114 | 4C | L | 108 | 01101100 | 154 | 6C | l |
| 13 | 00001101 | 015 | 0D | CR | 45 | 00101101 | 055 | 2D | - | 77 | 01001101 | 115 | 4D | M | 109 | 01101101 | 155 | 6D | m |
| 14 | 00001110 | 016 | 0E | SO | 46 | 00101110 | 056 | 2E | . | 78 | 01001110 | 116 | 4E | N | 110 | 01101110 | 156 | 6E | n |
| 15 | 00001111 | 017 | 0F | SI | 47 | 00101111 | 057 | 2F | / | 79 | 01001111 | 117 | 4F | O | 111 | 01101111 | 157 | 6F | o |
| 16 | 00010000 | 020 | 10 | DLE | 48 | 00110000 | 060 | 30 | 0 | 80 | 01010000 | 120 | 50 | P | 112 | 01110000 | 160 | 70 | p |
| 17 | 00010001 | 021 | 11 | DC1 | 49 | 00110001 | 061 | 31 | 1 | 81 | 01010001 | 121 | 51 | Q | 113 | 01110001 | 161 | 71 | q |
| 18 | 00010010 | 022 | 12 | DC2 | 50 | 00110010 | 062 | 32 | 2 | 82 | 01010010 | 122 | 52 | R | 114 | 01110010 | 162 | 72 | r |
| 19 | 00010011 | 023 | 13 | DC3 | 51 | 00110011 | 063 | 33 | 3 | 83 | 01010011 | 123 | 53 | S | 115 | 01110011 | 163 | 73 | s |
| 20 | 00010100 | 024 | 14 | DC4 | 52 | 00110100 | 064 | 34 | 4 | 84 | 01010100 | 124 | 54 | T | 116 | 01110100 | 164 | 74 | t |
| 21 | 00010101 | 025 | 15 | NAK | 53 | 00110101 | 065 | 35 | 5 | 85 | 01010101 | 125 | 55 | U | 117 | 01110101 | 165 | 75 | u |
| 22 | 00010110 | 026 | 16 | SYN | 54 | 00110110 | 066 | 36 | 6 | 86 | 01010110 | 126 | 56 | V | 118 | 01110110 | 166 | 76 | v |
| 23 | 00010111 | 027 | 17 | ETB | 55 | 00110111 | 067 | 37 | 7 | 87 | 01010111 | 127 | 57 | W | 119 | 01110111 | 167 | 77 | w |
| 24 | 00011000 | 030 | 18 | CAN | 56 | 00111000 | 070 | 38 | 8 | 88 | 01011000 | 130 | 58 | X | 120 | 01111000 | 170 | 78 | x |
| 25 | 00011001 | 031 | 19 | EM | 57 | 00111001 | 071 | 39 | 9 | 89 | 01011001 | 131 | 59 | Y | 121 | 01111001 | 171 | 79 | y |
| 26 | 00011010 | 032 | 1A | SUB | 58 | 00111010 | 072 | 3A | : | 90 | 01011010 | 132 | 5A | Z | 122 | 01111010 | 172 | 7A | z |
| 27 | 00011011 | 033 | 1B | ESC | 59 | 00111011 | 073 | 3B | ; | 91 | 01011011 | 133 | 5B | [ | 123 | 01111011 | 173 | 7B | { |
| 28 | 00011100 | 034 | 1C | FS | 60 | 00111100 | 074 | 3C | < | 92 | 01011100 | 134 | 5C | \ | 124 | 01111100 | 174 | 7C | \| |
| 29 | 00011101 | 035 | 1D | GS | 61 | 00111101 | 075 | 3D | = | 93 | 01011101 | 135 | 5D | ] | 125 | 01111101 | 175 | 7D | } |
| 30 | 00011110 | 036 | 1E | RS | 62 | 00111110 | 076 | 3E | > | 94 | 01011110 | 136 | 5E | ^ | 126 | 01111110 | 176 | 7E | ~ |
| 31 | 00011111 | 037 | 1F | US | 63 | 00111111 | 077 | 3F | ? | 95 | 01011111 | 137 | 5F | _ | 127 | 01111111 | 177 | 7F | DEL |

# Hello World!

```java
class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello, World!");
    }
}
```

# Operators



| Operator | Description | Example |
|---|---|---|
| + | Adds two operands | A+B will give 5 |
| - | Subtract second operand from first operand | A-B will give 1 |
| * | Multiply two operands | A*B will give 6 |
| / | Divide numerator by denumerator | A/B will give 1 |
| % | Divide numerator by denumerator and gives remainder | A%B will give 1 |
| ++ | Increase the operand by 1 | A++ will gives 4 |
| -- | Decrease the operand by 1 | A-- will gives 2 |

# Declaration, initialization, and assignment

- **Declaration:** int Number;
- **Initialization:** Number = 0;
- **Assignment:** Number  = 4;

# Explicit and implicit type conversion (casting)

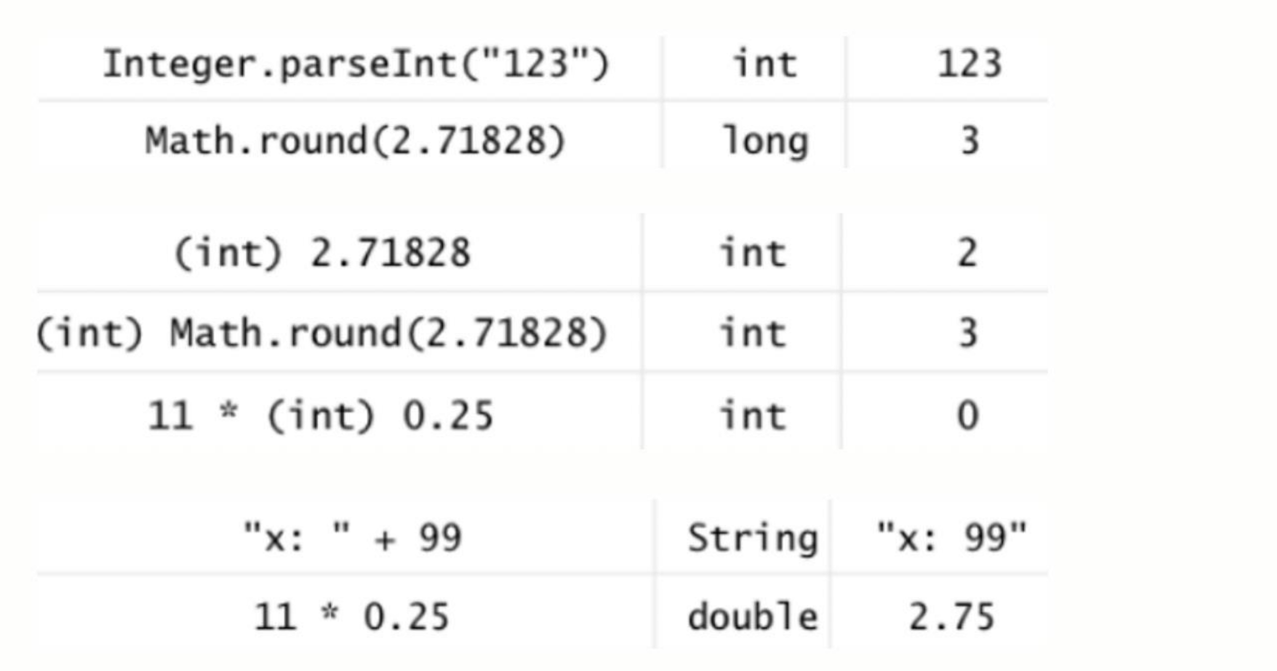
**implicit**

```
int a = 3;
double b=a;
```

**Explicit**

```
double a = 3.14;
Int b=a;
```

# Type conversion



| | | |
|---|---|---|
| Integer.parseInt("123") | int | 123 |
| Math.round(2.71828) | long | 3 |

| | | |
|---|---|---|
| (int) 2.71828 | int | 2 |
| (int) Math.round(2.71828) | int | 3 |
| 11 * (int) 0.25 | int | 0 |

| | | |
|---|---|---|
| "x: " + 99 | String | "x: 99" |
| 11 * 0.25 | double | 2.75 |

# Input

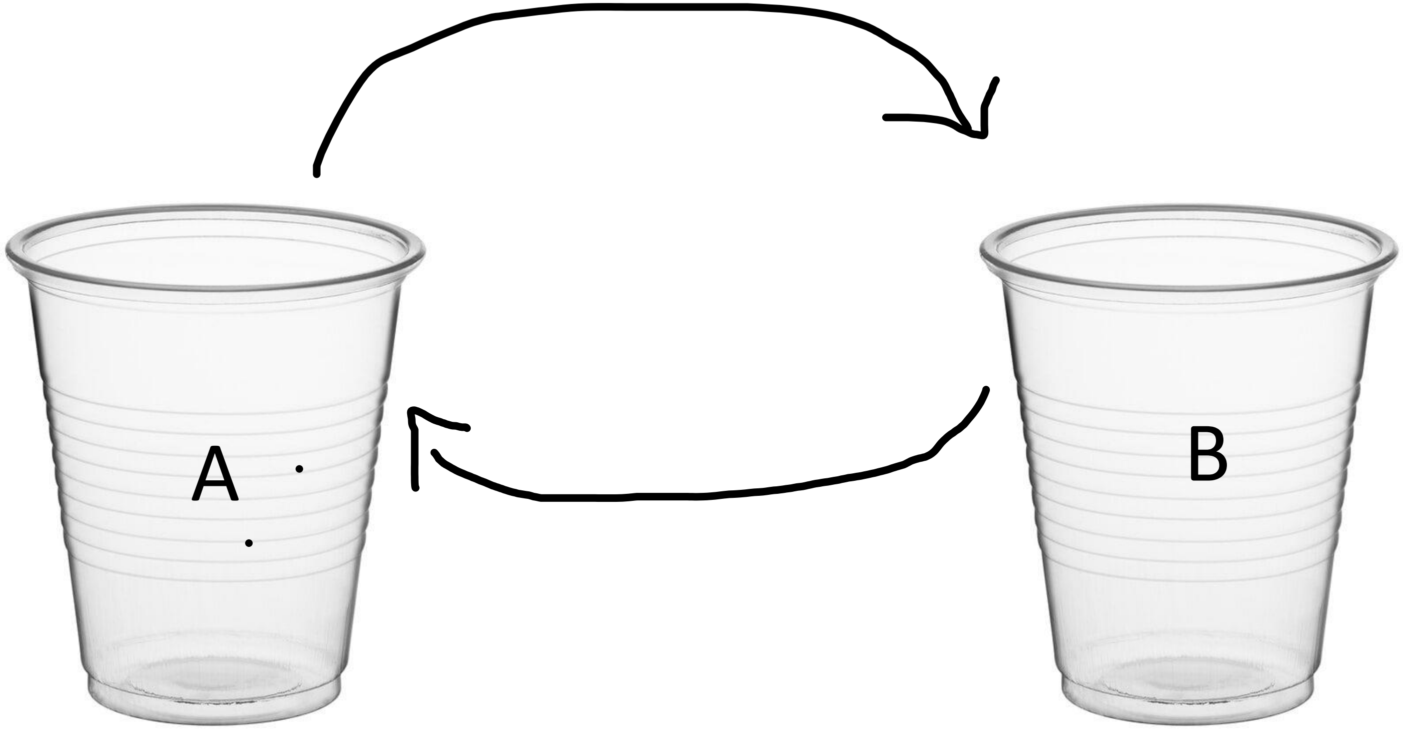
```java
import java.util.Scanner;

class Main {
  public static void main(String[] args) {
    Scanner Input = new Scanner(System.in);

    String Name = Input.nextLine();
  }
}
```

# Exercise 1:

- Exchange values in two variables



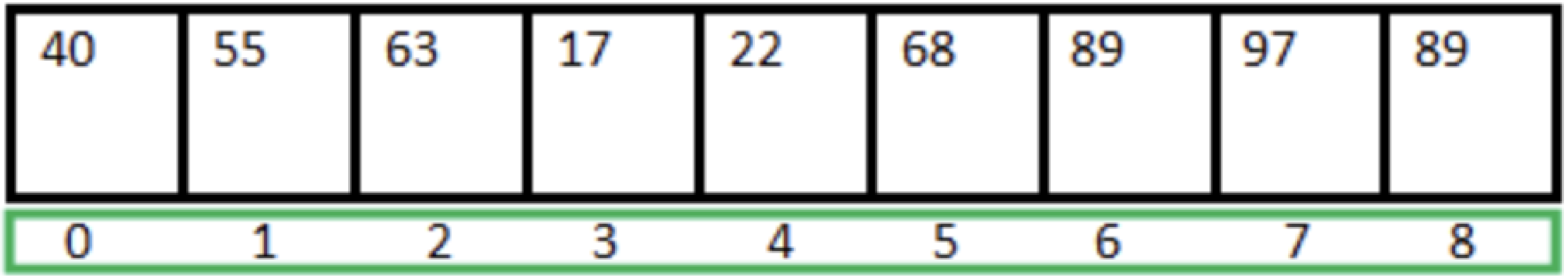
# Strings

```
String name1 = "Anna";
String name2 = "Anna";
```

# Strings

```
String Anna = new String("ATA");
String Hannah = new String("ATA");
```

# Arrays



| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 40 | 55 | 63 | 17 | 22 | 68 | 89 | 97 | 89 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |

# Arrays

```
int[] Numbers = {10, 20, 30, 40};
int[] Numbers = new int[4] ;
```

# Arrays

```
int[] numbers= new int[5];
System.out.println(numbers[0]);
```

**What will this print out?**

# Boolean conditions

- Boolean operators: ||, &&, ==, <=, >=, < ,>

# Conditions

- If, else, else if

```
if (condition){
Block of code
}
else If (condition){
Block of code
}
else {
Block of code
}
```

# Loops

```
while (1==1){
System.out.println("Hello!");
}
```

# Loops

```
do{
System.out.println("Hello")
}while(1!=1);
```

# Loops

for (initialization; condition; increment){

*Block of code*

}

# Loops

```
for (int i=0; i<N; i++){
System.out.println(MyArray[i]);
}
```

# Remove the maximum element in an array

- Find the largest element in an array and remove it

# Number reversal

- Given an input that is a number reverse the number such that a number 1234 becomes 4321

# Number reversal

```
Scanner Input = new Scanner(System.in);
int number = Input.nextInt();
int number_reduced=number;
int digits=0;
while (number_reduced>=1) {
number_reduced/=10;
digits++;
}
int reversed=0;
while (digits>=1) {
reversed+=number%10*Math.pow(10,digits-1);
digits--;
number=number-1;
}
System.out.println(reversed);
```

# Check if two strings are anagrams

- Check if two strings contain same letters but in a different order
- Examples:

  **silent** and **listen**

  Output = true

  **Shanghai** and **shanghai**

  Output = false