Contents

[**Overview**](#_ql0pw9y540iy) **2**

[**Comment  
(It's the same!)**](#_edd95cmpnc) **2**

[**Prompt for input**](#_3aa089rmfxpr) **3**

[**Variables and Data Types**](#_rbll1v20uuii) **3**

[**Output (print)**](#_lf5shlen9r70) **4**

[**Operators**](#_5ua4hv6uqb1y) **4**

[**if … else …   
(It's the same!)**](#_j680zvt8dc01) **5**

[**and, or, not   
(It's the same!)**](#_2whtsa8anaop) **5**

[**while loop  
(It's the same!)**](#_9317dh1p75di) **6**

[**do … while loop  
(It's the same!)**](#_sh14ipp61580) **6**

[**for loop  
(It's the same!)**](#_osd3ug25kces) **6**

[**Strings**](#_6ynsngr33aq6) **7**

[Formatted Strings](#_iemnmagr1ptw) 7

[Concatenation](#_2xiaqmxdvwfv) 7

[Length](#_xz086abqpe9u) 7

[Getting first character](#_umwa5ukx1q1w) 7

[Looping through every character](#_bb0zm0mo07k1) 8

[To uppercase/lowercase](#_qvqe24cafdy0) 8

[Search for position of a specified text within a String](#_vkc5eglia0li) 8

[String slicing](#_7o5pqg703ywb) 8

[**Functions**](#_459zfh4xemrk) **9**

# 

# Overview

This document shows equivalent code in C and Java.

Note that this document is

* NOT comprehensive
* NOT detailed

**"main"**

| **C** | **Java** |
| --- | --- |
| **int main(void) {**  int x = 2;  **}** | // a method needs to be "wrapped"  // in a class  **public class Test {  public static void doNothing() {** // nothing **}**  **public static void main(String[] args) {**  int x = 2;  **} }** |

**function/method**

| **C** | **Java** |
| --- | --- |
| int do\_nothing() {  return 1; }  int main(void) {  int result = do\_nothing();  } | // it's called a method  public class Test {  // Note:   // 1. include "public static"   // 2. inside a class definition  public static int doNothing() {  return 1;  }  public static void main(String[] args) {  int result = doNothing();  } } |

# 

# Comment *(It's the same!)*

| **C** | **Java** |
| --- | --- |
| **//** single line comment  **/\***  This is a multiline  comment.  **\*/** | **//** single line comment  **/\***  This is a multiline  comment.  **\*/** |

# Prompt for input

| **C** | **Java** |
| --- | --- |
| #include <stdio.h>  int main(void) {    // string value  char s[100];  printf("Enter s:");  scanf("%s", s);  // int value  int n;  printf("Enter n:");  scanf("%d", &n);    // double value  int m;  printf("Enter m:");  scanf("%lf", &m);  } | import java.util.\*;  public class Test {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  // string value  System.out.println("Enter s:");  String s = sc.nextLine();  // int value  System.out.println("Enter n:");  int n = sc.nextInt();  // double value  System.out.println("Enter m:");  double m = sc.nextDouble();    }  } |

# 

# Variables and Data Types

| **C** | **Java** |
| --- | --- |
| // many types  int counter3 = 100;  long counter4 = 1;  /// floating point  float miles1 = 1000;  double miles2 = 1000.0;  // strings are char array char name1[500]= "John"; // mutable  char \*name2 = "Bill"; // immutable  // char  char ch = 'a';  // boolean. True/False  // need to include <stdbool.h>  **bool** is\_free = true;  // an unset reference  //(i.e. a reference to nothing)  String empty = NULL; | //Only 4 integral types  byte counter1 = 1;  short counter2 = 1;  int counter3 = 1;  long counter4 = 1;  // Only 2 floating point  float miles1 = 1000.0f;  double miles2 = 1000.0;  // strings. **Note:**capital S String name = "John";  // Character Note: single quotes  char c = 'a';  // boolean type.  // true/false: lowercase  **boolean** isFree = true;  // an unset reference  //(i.e. a reference to nothing)  String empty = null; |

# Output (print)

| **C** | **Java** |
| --- | --- |
| // need to  // #include <stdio.h>  **printf**("apple\n");  **printf**("orange"); | // need to  import java.util.\*;  **System.out.println**("apple");  **System.out.print**("orange "); |

# 

# Operators

| **C** | **Java** |
| --- | --- |
| // integer division.  int answer = 3 / 2;  // normal division  double answer2 = 3.0 / 2;  // alternatively ..  // int/int returns an int hence  answer2 = (float)3 / 2;  int x = 2; x++; # post-increment  # += operator is also available  // need to  // #include <math.h>  double answer = pow(2, 3); | // integer division.  int answer = 3 / 2;  // normal division  double answer2 = 3.0 / 2;  // alternatively ..  // int/int returns an int hence  answer2 = (float)3 / 2;  x = 2; x++; // post-increment  // += operator is also available  x = Math.pow(2, 3); |

# if … else … *(It's the same!)*

| **C** | **Java** |
| --- | --- |
| if (t < 10) {  printf("Cheap\n");  } else if (t < 20) {  printf("Fair\n");  } else  // for 1-liner, curly brace is  // a gd practice  // (though not mandated)  printf("Expensive!\n");  int x = 1;  **// false if x == 0. Otherwise, true**  if (x) {  printf("OMG\n");  } | // no difference if (t < 10) {  System.out.println("Cheap"); } else if (t < 20) {  System.out.println("Fair");  } else  System.out.println("Expensive!");  int x = 2;  // Java treats boolean as a separate  // data type if (x > 0) {  System.out.println("OMG"); } |

# and, or, not *(It's the same!)*

| **C** | **Java** |
| --- | --- |
| if (x > 3 **&&** x < 10) {  printf("apple\n"); }  if (x < 3 **||** x > 10) {  printf("orange\n"); }  bool i\_raining = true;  if (**!**is\_raining) {  printf("go swimming\n"); } | // no difference  if (x > 3 **&&** x < 10) {  System.out.println("apple"); }  if (x < 3 **||** x > 10) {  System.out.println("orange"); }  boolean isRaining = true;  if (**!**isRaining) {  System.out.println("go swimming"); } |

# 

# while loop *(It's the same!)*

| **C** | **Java** |
| --- | --- |
| int x = 1;  while (x <= 5) {  printf("%d\n", x);  x++;  } | // no difference  int x = 1;  while (x <= 5) {  System.out.println(x);  x++;  } |

# do … while loop *(It's the same!)*

| **C** | **Java** |
| --- | --- |
| // no difference  int x = 1; do {  printf("%d\n", x);  x++;  } while(x <= 5); | // no difference  int x = 1; do {  System.out.println(x);  x++;  } while(x <= 5); |

# 

# for loop *(It's the same!)*

| **C** | **Java** |
| --- | --- |
| for (int i = 1; i < 3; i++) {  printf("%d\n", i);  } | // no difference  for (int i = 1; i < 3; i++) {  System.out.println(i);  } |

# 

# 

# Strings

## Formatted Strings

| **C** | **Java** |
| --- | --- |
| printf("%d %s\n", 2, "cow"); | /\* Reference: <https://www.cs.colostate.edu/~cs160/.Summer16/resources/Java_printf_method_quick_reference.pdf>  \*/  // Uses %n (instead of \n in C)  System.out.format("%d %s%n", 2, "cow"); |

## Concatenation

| **C** | **Java** |
| --- | --- |
| char output[256];  snprintf(output, sizeof(output),  "%d %s", 2, " cows"); | // primitive type + string is possible  int quantity = 10;  String message = quantity + " apples"; |

## Length

| **C** | **Java** |
| --- | --- |
| char \*word = "apple";  printf("%lu\n", strlen(word)); | String value = "abc"; int n = **value.length()**; // method |

## Getting first character

| **C** | **Java** |
| --- | --- |
| char \*word = "apple";  printf("%c\n", word[0]); | String value = "abc";  // char is a primitive type  char ch = **value.charAt(0);** |

## Looping through every character

| **C** | **Java** |
| --- | --- |
| char \*word = "apple";  for (int i = 0; i < strlen(word); i++) {  printf("%c\n", word[i]);  } | String value = "abc"; for (int i = 0; i < value.length(); i++){  System.out.println(value.charAt(i))  } |

## To uppercase/lowercase

| **C** | **Java** |
| --- | --- |
| // need to do it char by char printf("%c\n", toupper('c'));  printf("%c\n", tolower('C')); | String value = "abc"; value = value.toUpperCase(); value = value.toLowerCase(); |

## Search for position of a specified text within a String

| **C** | **Java** |
| --- | --- |
| // returns a pointer (not position) char str[] ="apple orange";  char \* substring;  substring = strstr (str,"orange"); | int result = "I love you!".**indexOf**("love");  if ( result == -1 ) {  System.out.println("Not found");  } else {  System.out.println(result); //output 2  } |

## String slicing

| **C** | **Java** |
| --- | --- |
| char \*two\_words = "apple orange";  char substring[7];  strncpy(substring, &two\_words[6], 6);  printf("%s\n", substring); | String value = "abcdef";  String partial = value.**substring**(3, 5); |

# 

# Functions

| **C** | **Java** |
| --- | --- |
| char \* do\_magic(int a,  int b) {  if (a > b){  return "apple";  }  return "orange";  } | // public to make it accessible  // by other classes  public static String do\_magic(int a, int b) {  if (a > b) {  return "apple";  }  return "orange"; } |

## 