Cheatography

Java Basics Cheat Sheet

by turback via cheatography.com/210084/cs/45280/

Variables

```
[ data_type ] variable_name;
[ data_type ] variab le_name = initia l_v alue;
int num = 5;
double value;
value = 22.4;
```

If Statement

```
if ( condition ) {
        sta tements
} else if ( condition ) {
        sta tements
} else {
        sta tements
}
```

Switch Statement

```
switch ( condition ) {
       case value:
                sta tements
                break;
       case value2:
                sta tements
                break;
        def ault:
                sta tements
                break;
}
```

For loop

```
for ( initialize ; condition ; increment ) {
        sta tements
}
```

While Loop

```
while ( condition ) {
        sta tements
}
```

Do-While Loop

```
do {
        sta tements
} while ( condition );
```

1D Arrays

```
element_type[] variable_name;
elemen t_t ype[] variab le_name = new elemen -
t_type [ array_size ] ;
elemen t_t ype[] variab le_name = { value1,
value2, value3, ... };
elemen t_type name = variab le_name [ index ];
variab le_name [ index ] = value;
// example
int[] arr = new int[10];
double[] nums = {22.1, 50, 34};
int a = arr[0];
arr[0] = 53;
// iteration, full array, from start to end
for (int i = 0; i < arr.le ngth; i++) {
       int element = arr[i];
}
```

Methods

```
[modifiers] return-type name ( parameters ) {
        sta tements
}
// modifiers
[ access -mo difier ] [ static / abstract ] [
synchr onized ] [ final ]
// examples
public static void printN ame() {
        sta tements
}
public void showNu mbe rs(int a, int b) {
        sta tements
}
public int sum(int a, int b) {
        return a + b;
}
```

Main

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

Class Structure

```
public class name {
        members
        con str uctors
        methods
}
public class name {
        private type var1;
        private type var2;
        public name ( type var1, type var2) {
                thi s.var1 = var1;
                thi s.var2 = var2;
        }
        public return _type method1 ( type param1,
type param2 ) {
                sta tements
        }
        public return _type method2 ( params ) {
                sta tements
        }
}
```

Primitive Data Types

byte	integers	1 byte
short	integers	2 bytes
int	integers	4 bytes
long	integers	8 bytes
float	decimals	4 bytes
double	decimals	8 bytes
char	characters	2 bytes
boolean	boolean	1 bytes

Arithmetic Operators

num1 + num2	addition
num1 - num2	subtraction
num1 * num2	multiplication
num1 / num2	division
num1 % num2	modulus
num++	post increment
num--	post decrement
++num	pre increment
--num	pre decrement

Assignment Operators

num1 = value		assignment
num1 += num2	num1 = num1 + num2	addition
num1 -= num2	num1 = num1 - num2	subtraction
num1 *= num2	num1 = num1 * num2	multiplication
num1 /= num2	num1 = num1 / num2	division
num1 %= num2	num1 = num1 % num2	modulus

Comparison Operators

num1 == num2	Equal To
num1 != num2	Not Equal
num1 > num2	Greater Than
num1 < num2	Less Than
num1 >= num2	Greater Than or Equal to
num1 <= num2	Less Than or Equal to

Logical Operators

boolean1 && boolean2	Logical AND
boolean1 || boolean2	Logical OR
! boolean1	Logical NOT

Binary Operators

num1 | num2	Bitwise OR
num1 & num2	Bitwise AND
num1 ^ num2	Bitwise XOR
~num1	Bitwise Complement
num1 >> num2	Right Shift
num1 << num2	Left Shift

Binary Operators (cont)

num1 >>> num2	Unsigned Right Shift

Type Conversion

```
// type casting
double val = 25.2;
int a = (int) val; // will be 25
// to string conversion
int number = 24;
String str = String.va lue Of( num ber);
// or with string concat: " hello: " + number
// string to int
String str = " 25";
int val = Intege r.p ars eIn t(str);
// string to double
String str = " 25.5 ";
double val = Double.pa rse Dou ble (str);
```

Literals

```
// integers
int num1 = 34; // base 10 (decimal)
int num2 = 042; // base 8 (octal)
int num3 = 0x22; // base 16 (hexad ecimal)
int num4 = 0b0010 0010; // base 2 (binary)
long value = 221115 312 2345L;
// decimals
float value = 155.4f;
double value = 155.4;
double value = 1.554e2; // exponent form
// char, String
char a = 'A';
char a = '\u0021';
char a = 67;
String b = " Hello World";
```

Access Modifiers

public	Any code
private	Only code in the same class
protected	Only code in the same class, inheriting class or the same package
package-private (default, set no modifier)	Only code in the same package

Modifiers

final	Cannot be overriden or extended
static	Belongs to the class, rather than an instance
abstract (on method)	Declares the method as abstract, requiring no body
abstract (on class)	Declares the class as abstract, can contain abstract methods
synchr-onized	Holds monitor on class instance during method execution
transient	Attributes and methods are skipped when serializing the object containing them (via java object serialization)
volatile	The value of the attribute is not cached on CPU

Math Methods

```
// basic
double pi = Math.PI; // pi
double val = Math.a bs( -15); // 15, absolute
value
double val = Math.s ign um( -15); // -1, sign (-1,
0, 1)
double val = Math.m ax(12, 13); // 13, max of 2
numbers
double val = Math.m in(12, 13); // 12, min of 2
numbers
double val = Math.s qrt (16); // square root of
number
double val = Math.p ow(2, 4); // 2 to the power of
4
// round
double val = Math.c eil (15.2); // 16, round up
double val = Math.f loo r(1 5.2); // 15, round
down
```

Math Methods (cont)

> double val = Math.round(15.2); // 15, round to the closest
// trigo
double degrees = Math.toDegrees(Math.PI); // radians to degrees
double radians = Math.toRadians(180); // degrees to radians
double val = Math.sin(radians); // sine func
double val = Math.cos(radians); // cosine func
double val = Math.tan(radians); // tangent func
double radians = Math.asin(value); // arc sine func
double radians = Math.acos(value); // arc cosine func
double radians = Math.atan(value); // arc tangent func

Method Invocation

```
method_name();
method _name( parameters );
return _type name = method _name( parameters );
printN ame();
int sum = sumNum ber s(11, 90);
```

Objects

```
// creation
Class_Name var_name;
Class_Name var_name = new Class_ Name();
Class_Name var_name = new Class_ Name( parameters
);
Person person;
Person person = new Person ("Barak Obama", 25);
// access
type name = var_na me.a tt ribute;
var_na me.a tt ribute = value;
int a = person.age;
person.age = 55;
// method invocation
var_na me.m et hod _na me();
var_na me.m et hod _name( parameters );
return _type name = var_na me.m et hod _name(
parameters );
String name = person.ge tNa me();
person.wa lkD ist anc e(5);
```