Java Cheat Sheet

KEYWORDS

abstract	double	int	super
assert	else	interface	switch
boolean	enum	long	synchronized
break	extends	native	this
byte	final	new	throws
case	float	private	transient
class	goto	public	void
catch	finally	package	new
const	if	return	volatile
continue	implements	short	while
default	import	static	do
instanceof	strictfp*	true	false
if	while	for	

PRIMITIVE DATA TYPES

byte	8 bits
short	16 bits
int	32 bits
long	64 bits
float	32 bits
double	64 bits
char	16 bits
boolean	true/false (1/0)

ARITHMETIC OPERATORS

+	addition
-	subtraction
*	multiplication
/	division
%	remainder (modulus)
++var	preincrememnt
--var	predecrement
var++	postincrement
var--	postdecrement

RELATIONAL OPERATORS

<	less than
<=	less than or equal to
>	greater than
>=	greater than or equal to
==	equal to
!=	not equal to

LOGICAL OPERATORS

&&	and
||	or
!	not
^	exclusive or

Java Cheat Sheet

IF STATEMENT

```
if (condition){
    statements;
}


if (condition){
    statements;
}
else{
    statements;
}


if (condition1){
    statements;
}
else if(condition2){
    statements;
}
else{
    statements;
}
```

ASSIGNMENT OPERATORS

=	assignment (int = x;)
+=	addition assignment (x += y; (x + x = y;))
-=	subtraction assignment (x -= y;)
*=	multiplication assignment (x *= y;)
/=	division assignment (x /= y;)
%=	remainder assignment (x %= y;)

COMMENTS

```
//      single line comment
/**/    multi-line comment
```

SWITCH STATEMENT

```
switch(expression){
    case value:
        statements;
        break;
    …
    case value:
        statements;
        break;
    default:
        statements;
}
```

LOOP STATEMENT

```
while(condition){
    statements;
}

do{
    statements;
}while(condition);

for (init; condition; adjustment){
    statements;
}
```

Java Cheat Sheet

Java is Object Oriented (OOP – object oriented programming)
“Objects” are made of classes (keyword class)
Classes are a blueprint for an object (defines the “object”)
All programs start from a “main” function

CLASS EXAMPLE

```
public class Robot{         //define Robot a public class in Robot.class file
    private:               /*private class variables only accessible
        int xPosition;                      by class member functions*/
        int yPosition;
    public:                //public member functions to manip private data
        Robot(){}          //class constructor
      //return type – function name – parameters(if any)
        void setPositions(int a, int b){
            xPosition = a;  /*parameters “a” and “b” set the private members
            yPosition = b;    when this function is called*/
        }
        //set xPosition
        void setXposition(int a){
            xPosition = a;
        }
        //set yPosition
        void setYposition(int a){
            yPosition = a;
        }
        //return xPosition (return type of int)
        int  getXposition(){
            return xPosition;
        }
        //return yPositon
        int  getYposition(){
            return yPosition;
        }
}
```

MAIN.JAVA

```
import Robot.class;   //import the robot class
import System;        //import System to print to console
public class Main{
    public static void main(String[] args){ //main program function
        Robot robo = new Robot();   //instantiate new robot object
        robo.setPositions(32, 29);  //call setPositions() with dot operator
        //call get functions to print to console
        System.out.println + robo.getXposition() + robo.getYposition();
    }
}
```

Java Cheat Sheet

EXAMPLES

```
int someNum = 10;  //assign the integer value of 10 to “someNum”
int otherNum = 9;
float num = 7.02;  //assign the floating point value of 7.02 to “num”
float num2 = 2.20;
Boolean success = true; //assign the Boolean value of true to “success”
someNum++;  //new value of someNum: 11
otherNum--; //new value of otherNum: 8
float num3 = num + num2;  //value of num3: 9.22
num3 += num; //new value of num3: 16.24
```

```
if (success){ //if success true
    someNum--;
    robot.moveForward();
}
```

```
if (!success){ //if success NOT true
    someNum++;
    robot.moveBackwards();
}
```

```
switch(sensor.read()){ //assume sensor.read() returns an integer value
    case 1:
        motor1.move(“forward”);
        motor3.move(“reverse”);
        break;
    case 2:
        motor2.move(“forward”);
        motor4.move(“reverse”);
        break;
    case 3:
        motor1.move(“reverse”);
        motor3.move(“forward”);
        break;
    case 4:
        motor2.move(“reverse”);
        motor4.move(“forward”);
        break;
    default:
        robot.log(“error, error, crash and burn!”);
```