

Very Shortened JAVA Notes (class 9 icse)

Black White Purple Cyan Grey Light pink brown

Program - A set of instruction given to the computer to perform a particular task.

Instructions - Set of Data.

Data - Any piece of information, a collection of information

Programming language - A programming language is a formal language that allows programmers to write instructions or code to create computer programs.

It is a set of rules and syntax that provides a way to communicate with a computer system and instruct it to perform specific tasks or operations.

Object: An object is an instance of a class. It represents a particular entity or concept in the program. Objects have properties (data or attributes) and behaviors (methods or functions).

For example, if we have a class called "Car," an object of that class could be a specific car instance with its own unique characteristics and actions.

Class: A class is a blueprint or template that defines the properties and behaviors that objects of that class will have. It serves as a blueprint for creating multiple objects with similar characteristics. A class defines the attributes (data members) and methods (member functions) that objects will possess.

For example, a class called "Car" can define attributes like "color," "make," and "model," as well as methods like "start," "accelerate," and "stop."

State: The state of an object refers to the current values of its attributes. It represents the data associated with the object at a specific point in time.

For example, a car object's state could include its color, make, and current speed.

Behavior: The behavior of an object refers to the actions or operations that the object can perform. It is determined by the methods or member functions defined within the class.

For example, a car object's behavior could include methods like "startEngine()" or "accelerate(speed)."

Object Oriented Programming

Object-oriented programming (OOP) - is a programming paradigm that organizes software design around objects, which are instances of classes.

It is based on the concept of encapsulating data (attributes) and behaviors (methods) together into objects.

4 pillars of object oriented programming:

a) Abstraction: Abstraction involves representing complex systems by simplifying or abstracting their essential features, focusing on what is necessary for the current context.

b) Encapsulation: Encapsulation is the process of bundling data (attributes) and related behaviors (methods) into objects. It enables data hiding, as the internal state of an object is not directly accessible from outside.

c) Inheritance: Inheritance allows the creation of new classes (derived or child classes) that inherit properties and behaviors from existing classes (base or parent classes)

d) Polymorphism: Polymorphism means the ability to take different forms. In OOP, polymorphism allows objects of different classes to be treated as objects of a common base class.

Introduction To Java

Important Points

- Java was initially called Oak
- James Aurther Gosling is the father of Java
- Compilation is the process that converts source code to byte code.
- Java Virtual Machine is implemented in software and runs using the capabilities provided by your operating system and computer hardware.
- Java source code has .java extension
- Java byte code has .class extension
- Java is platform independent.

DATA Types

Data Types (**primitive**) : Remember the names of these

- **byte** – can store very small numbers **56, -66, 89** etc.
- **short** – some more numbers like **1024, -569** etc. but no decimal
- **int** – can store integers like **5, -53, 125566** etc. but no decimals.
- **long** – can store large number of integers like **1256689663, -88898526** etc.
- **float** – can store small number of decimal value like **5.256, -8.896** etc.
- **double** – can store large number of decimal value like **1.41428962**.
- **char** – one single character like inside ' ' like **'#'** **'J'** **'A'**.
- **Boolean** – can store **true** or **false**.

Any other data type is considered as **NON-PRIMITIVE**, also called **REFERENCE TYPE** datatype.

OPERATORS

Arithmetic Operators

+	Addition	$x + y$, $5 + 10$
-	Subtraction	$x - y$, $15 - 6$
*	Multiplication	$x * y$, $10 * 2$
/	Division	x / y , $45 / 9$
%	Modulus	Gives the remainder

Unary Operators

++m first add 1 then show value in same line | **m++** show value then add 1 from next line

--n first subtract 1 then show value in the same line | **n--** show value the subtract -1 from the next line

Assignment Operators

=	used to assign value to a variable, like	$x = 12$, $p = -12$, $a = b$ etc.
+=	add the value to given variable, like	$x = 10$ $x += 20$ so x becomes $10 + 20 = 30$
-=	subtract the value from variable, like	$x = 10$ $x -= 5$ so x becomes $10 - 5 = 5$
*=	multiply the value to variable, like	$x = 10$ $x *= 5$ so x becomes $10 * 5 = 50$
/=	divide the variable by the value, like	$x = 10$ $x /= 5$ so x becomes $10 / 5 = 2$

Relational Operators

>	Greater than	$10 > 2$, $0 > -1$
<	Less than	$-1 < 0$, $5 < 15$
>=	Greater than equals	
<=	Less than equals	
==	Equal to	$2 * 2 == 4$, $5 - 5 == 0$
!=	NOT Equal to	$2 * 2 != 10$

Logical Operators

&& and

|| or

! not

Mathematical Library Methods

> Pre defined functions

> In java.lang package

> 13 Methods/functions

> to use, simply say `maths.function(value);`

`abs()` - absolute value i.e, removes - sign. real life $|x|$, $|a|$, $|b|$ etc.

```
int x = Math.abs(-5);
```

```
x = 5
```

`sqrt()` - square root of a positive number.

```
int x = Math.sqrt(81);
```

```
x = 9
```

`cbrt()` - cube root of a positive number.

```
int x = Math.cbrt(27);
```

```
x = 3
```

`pow()` - power of a^b .

```
int x = Math.pow(2 , 3) ;    // 2 to-the-power 3
```

```
x = 8
```

`round()` - round of to the nearest integer.

```
int x = Math.round(5.5);
```

```
int x = 6
```

`rint()` - round of to the nearest integer with .0 at end

```
double x = Math.rint(4.5);
```

```
x = 5.0
```

`floor()` - nearest integer but less than the given number.

```
int x = Math.floor(9.8)      // 9 < 9.8 < 10. Choosing the lower value
```

```
x = 9
```

`ceil()` - nearest integer but greater than given number.

```
int x = Math.ceil(8.2)           // 8 < 8.2 < 9. Choosing the upper value
x = 9
```

`max()` - largest value b/w two numerical value.

```
double x = Math.max(5.3 , 1);
x = 5.3
```

`min()` - smallest value b/w two numerical value.

```
double x = Math.min(5.3 , -3);
x = -3
```

`random()` - random value b/w 0.0 and 1.0.

```
double x = Math.random()
x = 0.256347899 , 0.25796354 , 0.68564586 etc.
```

Conditional Statements In Java

Program can be two types

- Normal Flow
- Conditional Flow

Mainly 3 statements in conditional flow

`if()` ; `else if(){} ; else()`

How to use?

```
if(condition )
    System.out.println(" ");
else if (condition ) {
    System.out.println();
}
else
    System.out.println();
```

SWITCH CASE

NESTED IF STATEMENTS

To Start a Java Program

These two lines are a must know.

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

To Start a Java Program That requires input from user

When required user input, we need to use the java.util.scanner class.

```
import java.util.Scanner;  
public class practice {  
    public static void main(String[] args) {  
        Scanner scanner = new Scanner(System.in);  
    }  
}
```

- Next, Give a

System.out.println("What u need from the user: ");

- Store the data in a variable

Depending on what the data type of the input we need, we use next[Datatype]

Most commonly used ones are

scanner.nextInt();

scanner.nextLine();

scanner.nextDouble();

[NOTE]

When using nextInt() or nextDouble() use a nextLine to free up the scanner box.

Here's an extract example that takes the age of a person and prints it.

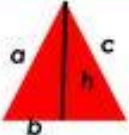
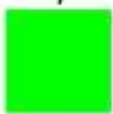


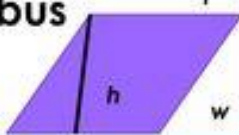
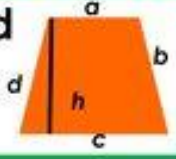


```
System.out.println("What is your age? ");  
int age = scanner.nextInt();  
scanner.nextLine();  
  
System.out.println("Your age is " + age);
```

MUST KNOW MATHS FORMULAE


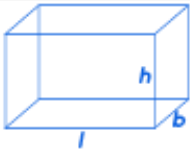






Awesome Teacher Resource



Geometric Formulas for Perimeter and Area

Shape	Perimeter	Area
Triangle 	$P = a + b + c$	$A = \frac{1}{2} (c \times h)$ C = the base
Square 	$P = 4 \times l$	$A = l \times l$
Rectangle 	$P = 2 \times (l + w)$	$A = l \times w$
Parallelogram 	$P = 2 \times (l + w)$	$A = b \times h$ b = the length
Rhombus 	$P = 2 \times (l + w)$	$A = b \times h$ b = the length
Trapezoid 	$P = a + b + c + d$	$A = \left(\frac{a+b}{2} \right) h$
Regular n-agon 	$P = 5a$ This is when all sides are equal	$A = \frac{1}{2} (h \times n \times a)$ n = the number of sides
Circle 	$P = 2 \pi r$	$A = \pi r^2$

Volume Formulas

Name of the Solid	Figure	Volume	Nomenclature
Cube		a^3	a : side of the cube
Cuboid		lbh	l : length b : breadth h : height
Cone		$\frac{1}{3}\pi r^2 h$	r : radius of the base h : height l : slant height
Cylinder		$\pi r^2 h$	r : radius of the base h : height
Sphere		$\frac{4}{3}\pi r^3$	r : radius
Hemisphere		$\frac{2}{3}\pi r^3$	r : radius
Prism		Area of base \times Height	-
Pyramid		$\frac{1}{3}$ (area of the base) \times height	-