Java Basic Syntax

Variables

Name	Wrapper Class	Default Value	Print format
byte	Byte	0	-
short	Short	0	%d
int	Integer	0	%d / %i
long	Long	0	%d
float	Float	0	%f
double	Double	0	%f
char	Character	'\u0000' (means 0 in ASCII)	%c
boolean	Boolean	FALSE	-
Set of characters	String	""	%s

Input and Output

Input

```
Scanner scan = new Scanner(System.in); //create Scanner

System.out.print("Enter username: ");
String username = scan.nextLine();
```

Output

```
System.out.println("Print with new line");
System.out.print("Print without new line");
System.out.printf("Print with format %s %d", "James", 3000);
```

Repetition

While Loop

```
while(condition) {
  //do something
}
```

For Loop

```
for(initialization; condition; result) {
  //do something
}
```

For each loop

```
for(type variable : arrayName) {
  //do something
}
```

Selection

If Else

```
if (condition) {
  //do something
} else if(condition) {
  //do something
} else {
  //do something
}
```

Ternary

```
variable = (condition) ? "True" : "False";
```

Switch Case

```
switch(variable) {
case condition:
  //do something
  break;
default:
  //do something
}
```

String functions

Function	Description	Result
.equals(s)	check if the string is equals with string inside the parameter	boolean
.length()	get number of characters inside the string	int
.toLowerCase()	convert the string into lowercase	string
.toUpperCase()	convert the string into uppercase	string
.charAt(i)	get a character from the string at specified index	char
.startsWith(s)	check if the string is starts with string inside the parameter	boolean
.endsWith(s)	check if the string is ends with string inside the parameter	boolean
.indexOf(s)	get the starting index of the string inside the parameter	int
.equalsWithIgnoreCase(s)	get the starting index of the string inside the parameter	int

Operator

Arithmetic Operator

Operator	Description	Example
+ (addition)	Add values on either side of the operator	A = 10, B = 20 A+B will give 30
- (subtraction)	Subtracts right-hand operand from left-hand operand	A = 20, B = 10 A-B will give 10
* (multiplication)	Multiplies values on either side of the operator.	A = 20, B = 10 A*B will give 200
/ (division)	Divides left-hand operand by right-hand operand	A = 20, B = 10 A/B will give 2
% (modulus)	Divides left-hand operand by right-hand operand and	A = 22, B = 10 A%B will give 2
++ (increment)	Increases the value of operand by 1	A = 1 A++ will give 2
-- (decrement)	Decreases the value of operand by 1	A = 1 A-- will give 0

Relational Operator

Operator	Description	Example
== (equal to)	Checks if the values of two operands are equal or not	A = 10, B = 20 A == B -> false
!= (not equal to)	Checks if the values of two operands are equal or not	A = 20, B = 10 A != B -> true
> (greater than)	Checks if the value of left operand is greater than the	A = 20, B = 10 A > 10 -> true
< (less than)	Checks if the value of left operand is less than the value	A = 20, B = 10 A < B -> false
>= (greater than or equal to)	Checks if the value of left operand is greater than /	A = 10, B = 10 A >= B -> true
<= (less than or equal to)	Checks if the value of left operand is less than / equal to the value of right operand	A = 10, B = 10 A <= B -> true

Array / ArrayList / Vector

Array

```
String[] names = {"Andy", "Chris", "John"};

String name = names[0];
```

ArrayList / Vector

```
ArrayList<String> names = new ArrayList<String>();
names.add("John");
names.add("Chris");

Vector<String> names2 = new Vector<String>();
names2 .add("John");
names2 .add("Chris");
```

Difference between ArrayList and Vector

- ArrayList and Vector both implements List interface and maintains insertion order

- But there are many differences between ArrayList and Vector classes that are given below.

ArrayList	Vector
1) ArrayList is not synchronized	Vector is synchronized
2) ArrayList increments 50% of current array size if number of element exceeds from its capacity	Vector increments 100% means doubles the array size if total number of element exceeds
3) ArrayList can only use Iterator	Vector can use Iterator and Enumeration to traverse over the elements
4) ArrayList is a parts of the Collection framework and was introduced in JDK	Vector is present in the earlier versions of Java as a legacy class