Java Basics

→ 20 questions

6 months ⇒ 100 + 5

→ 900
− 100
800

→ Consistent → 5 coding

sheets
15 April → 5 →
06 April → 6 →
0 → ——

→ No doubt / question is silly.

→ 5 CW + 5 H.W → 10 Question

→ Test

→ ~~000 000~~ Revision → Basic Java → Revision Day.

Topics

- Print in Java
- Datatypes and Variables
- Operators

Printing in Java

→ 'Welcome to bootcamp'

→ Hello everyone

↪ System.out.println

↪ System.out.print

→ System.out.println("Welcome to bootcamp");

→ System.out.print("Welcome to bootcamp");

→ system.out.print → Print in a single line

→ System.out.println → Print in multiple lines

```
System.out.print("abc");
System.out.print("def");
```

→ abcdef

→ def

(a)
```
System.out.print("abc");
System.out.print("def");
```
o/p abcdef

(c)
```
System.out.println("abc");
System.out.print("def");
```
abc
def.

(b)
```
System.out.println("abc");
System.out.println("def");
```
o/p
abc
def

(d)
```
System.out.print("abc");
System.out.println("def");
```
abc def

(a)
(b)
(c)
(d)

System.out.println

↳ System.out.print

\n , \t → Escape character

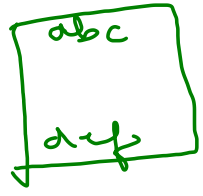
move to next line

```
System.out.println("abc\n");
System.out.println("def");
```

O/P
abc
def

```
System.out.println("abc\n");
System.out.println("def");
```

O/P ...



***** → 5 stars.

___* → 1 star 3 spaces

__* → 1 star 2 spaces

_* → 1 star 1 space.

***** → 5 stars.

1 tab → 4 | 5 space

"3\t" 3 ———

1st line

*****

↳ System.out.println("*****");

2nd line ___*

↳ System.out.println("___*");

3rd line

↳ System.out.println("__*");

4th line

↳ System.out.println("_*");

5th line

↳ System.out.println("*****");

Variables and datatype → what type "info" variable is storing

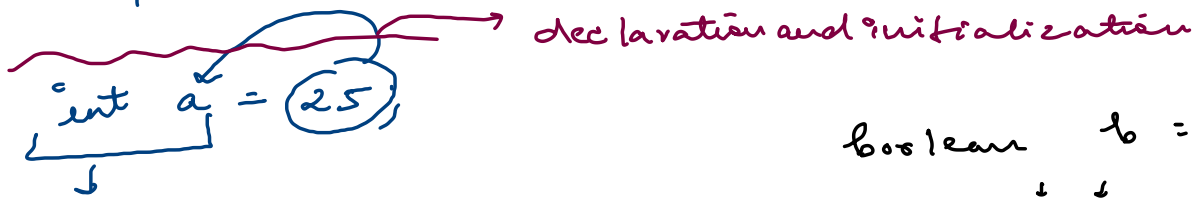
↳ Containers to store the data

Syntax

datatype variable-name = value ; (value: optional)

str → double quotes
Character → single quote.

Q: Integer info in variable-name 'a' and value is 25;



boolean b = true | false;

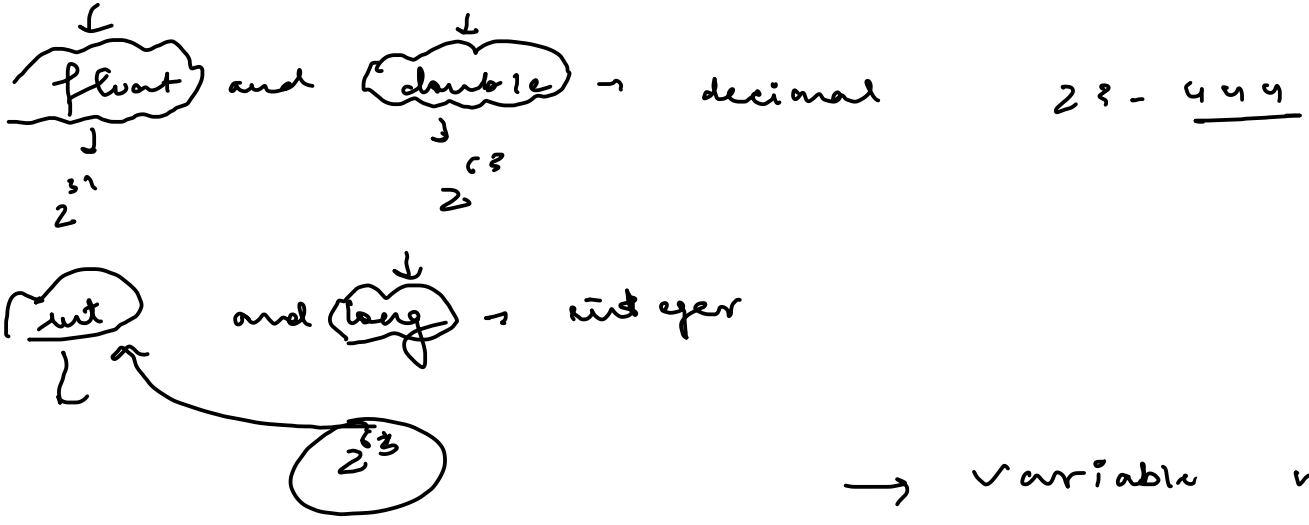
char ch = 'a'; char ch = a ✗

double d = 23.55;

↳ primitive datatype and non-primitive datatypes

long { ↳ int → integer
↳ char → character
↳ Boolean → true | false
↳ double | float → decimal

↳ String → String str = "Nanyuene"



→ variable name starts with either english alphabets or special ($, _) ;

→ Difference is only in the range

→

int a = 65;

The value of a is: 65

System.out.println("The value of a is:" + a)

Note:- Whenever we want to print some message and variable data also use '+' symbol.

Taking user input → Scanner class

int a = 65;

int b = 100;

import java.util.Scanner; // → This way is used to import a single class.

1) import java.util.*; // Include all classes that comes under java.util.

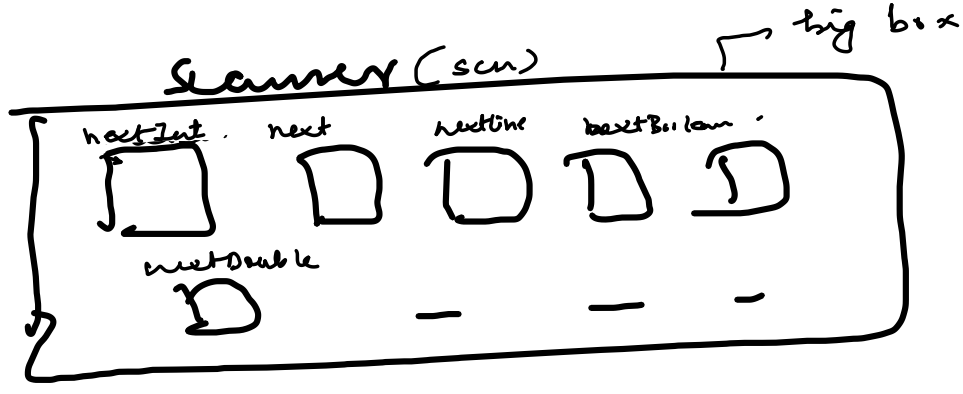
2) → Scanner scn = new Scanner (System.in);

scn → object name

3) Learn it for now

Scanner scn = new Scanner(System.in);

int → int a = scn.nextInt(); →



boolean → boolean b = scn.nextBoolean();

double → double d = scn.nextDouble();

String
- String str = scn.next();
- String str = scn.nextLine();

char
- char ch = scn.next().charAt(0); → String input + character present at 0th index
- char ch = scn.nextLine().charAt(0);

Navgurukul Bootcamp

Int -

```java
import java.util.*;

public class Main {
    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int a = scn.nextInt();

        System.out.println("The value of a is: " + a);
    }
}
```

Boolean

```java
import java.util.*;

public class Main {
    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        boolean a = scn.nextBoolean();

        System.out.println("The value of a is: " + a);
    }
}
```

double -

```java
import java.util.*;

public class Main {
    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        double a = scn.nextDouble();

        System.out.println("The value of a is: " + a);
    }
}
```

Ex: NavGurukul_Bootcamp

NavGurukul

String

↳ String str = scn.next(); → stop reading the data once it gets whitespace -

↳ String str = scn.nextLine();

Ravi

char →

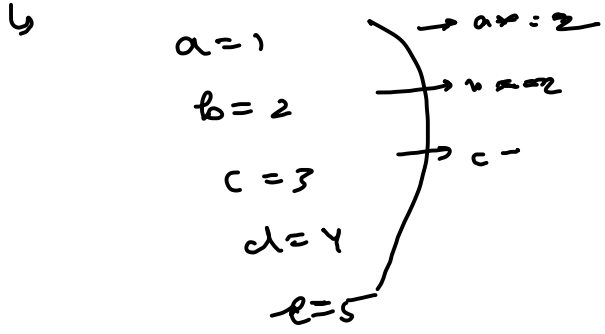
1) scn.next().charAt(0) → R

scn.nextCharacter() ~~X~~

2) scn.nextLine().charAt(2) → v.

String

Find product



int ans = a*b*c*d*e;

Syso (ans);

a+b+c+d+e;

```java
public class Main {
    public static void main(String[] args) {
        int a = 34;
        boolean b = true;
        char ch = 'q';
        double d = 23.44;
        String str = "NavGurukul";

        System.out.println(a);
        System.out.println(b);
        System.out.println(ch);
        System.out.println(d);
        System.out.println(str);
    }
}
```

Operators in Java

↳ Operators are used to perform certain actions on the data/variable.

int ans = one (1) + two (2) + three (3) + four;

one = one * 2

↳ (a) Arithmetic Operator → +, −, *, %, /

↳ (b) Assignment Operator

↳ (c) Comparison Operator

↳ (d) Logical Operator

↳ (e) Bitwise operator

↳ Bits

10

(a) Arithmetic operator

+, −, *, /, %

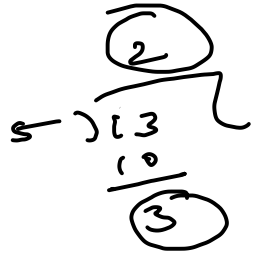
int a = b + c ; → addition

e = a * b ; → multiplication

f = a − b ; → subtraction

g = a / b ; → Quotient

h = a % b ; → Remainder



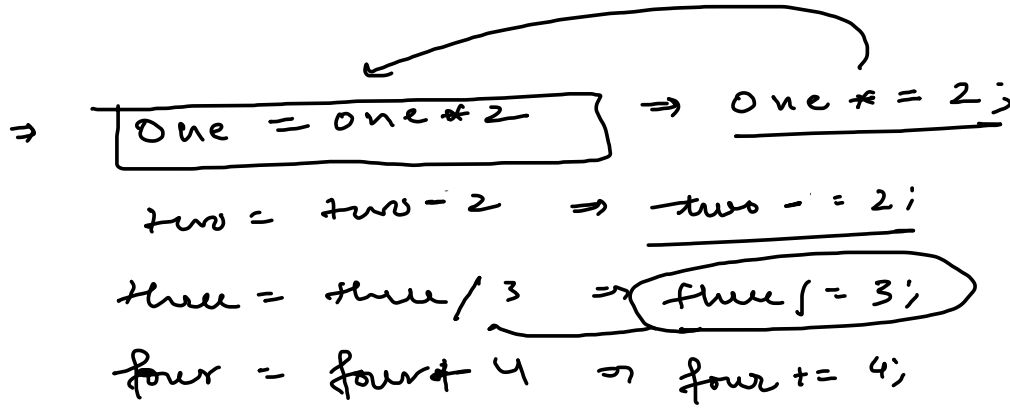
13 / 5 = (2)

13 % 5 = (3) ← Remainder

(6) Assignment Operator (=)

int a = 10

assignment operator



one *= 5 ⇒ one = one * 5;

(5) Comparison Operator → O/P → boolean

>, >=, <, <=, !=, ==

(> : greater; == : equal)

int a = 10, b = 50;

a > b → true;

④ Logical operator → O/P → boolean

↳ `&&` , `||` , `!`

`&&` → AND

`||` → OR

`!` → NOT

AND → True → when all the condⁿ are true

↳ False → when atleast one condⁿ is false

OR → True → when atleast one cond is true

↳ False → when all the conds are false.

NOT → used to flip the result

```
int a = 50, b = 70
cond1 ⇒ a > b      ⇒ F
cond2 ⇒ a == 50    ⇒ T
cond3 ⇒ b != 50    ⇒ F
```

⇒ False

cond1 && cond2 && cond3 → false

cond1 || cond2 || cond3 → true

↳ vote

↳ age >= 18 and Indian

age >= 18 && country == "India"

Conditional Statements

cond1 , cond2, cond3

↳ You want to take decision on the basis on cond^n

↳ cond1 → true → "Hello everyone"

↳ print → Invalid .

↳ if , else , else if

if

int a = 5

Cond1 → (a == 5) → "Welcome to bootcamp"

Syntax :- if (condition) {

}

→

else →

Syntax else {

}

```java
import java.util.Scanner;
public class Solution
{
    public static void main(String[] args) {
      int a = 50, b= 80;
      if (a == 50 && b == 70) {
        System.out.println("Welcome to Bootcamp");
      } else {
        System.out.println("Invalid condition");
      }

    }
}
```

else if

↳

cond1 → age > 40 → print (Older)

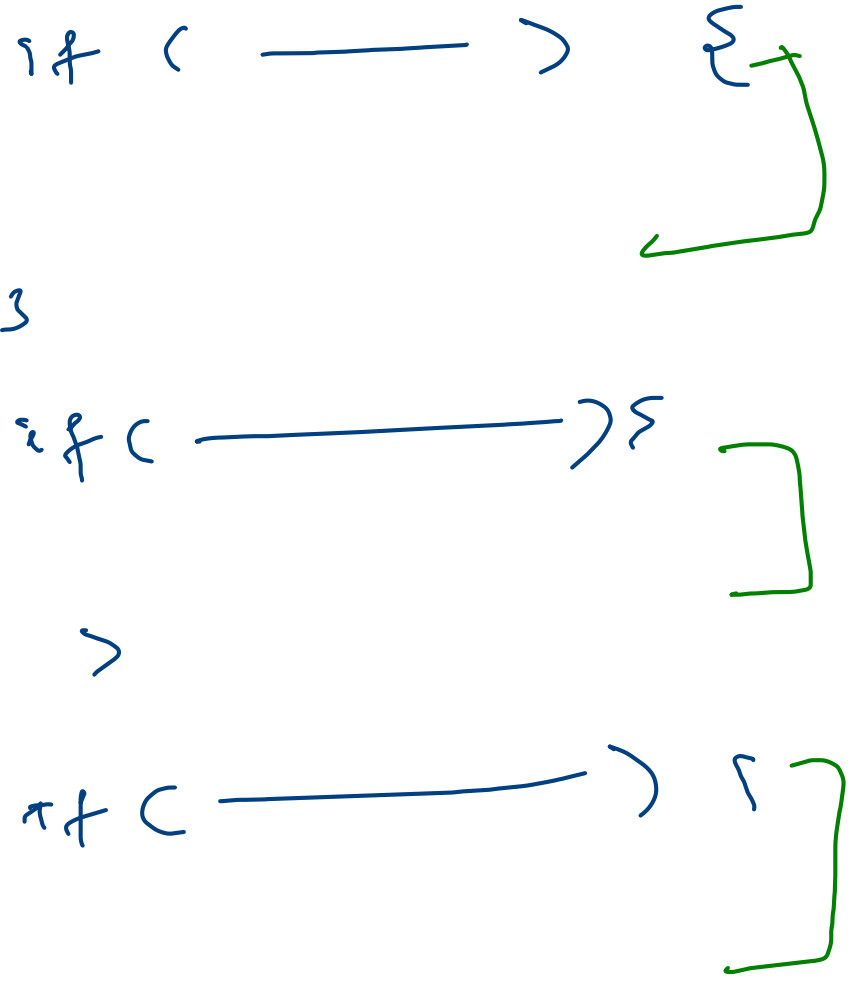
cond2 → age > 20 and age <= 40 → print (——)

cond3 → age < 20 → print (——)

if → in a single logic, we can use only one if statements

else if → ——— x ——— , ——— x multiple else if — x ——

```
if (age > 40) ————————
else if (age > 20 and age <=40 ) ————————
else ————————
```



Slump Discount

↳ lb → units

1 unit = 100

totalcost = 0.1 → decimal

↳ decimal → double

b = 7

700 > 1000 ⇒ 10%

$\frac{10}{100} = 0.1$

a = 700

cut - double

b = 20 ⇒ 2100 > 1800 ⇒ 2000 - $\frac{2000 \times 10}{100}$

⇒ 1800

Grading system

↳ marks = 92

↳ multiple conditions on a same logic

↳ marks > 90 → excellent

↳ marks > 80 and marks <= 90 → good

↳ marks > 70 and marks <= 80 → fair

↳ marks > 60 and marks <= 70 → met expectations

↳ marks <= 60 → below par.

if (marks > 90) print (———)

elseif (marks > 80 && marks <= 90)

elseif (marks > 70 && marks <= 80)

elseif (marks > 60 && marks <= 70)

else print (———)

print (———)

print (———)

print (———)

```java
int marks = scn.nextInt();

if (marks > 90) {
    System.out.println("excellent");
} else if (marks > 80 && marks<=90) {
    System.out.println("good");
} else if (marks > 70 && marks<=80) {
    System.out.println("fair");
} else if (marks > 60 && marks<= 70) {
    System.out.println("meets expectations");
} else {
    System.out.println("below par");
}
}
```

marks > 90

```
int marks = scn.nextInt();

if (marks > 90) {
    System.out.println("excellent");
} else if (marks > 80 && marks <= 90) {
    System.out.println("good");
} else if (marks > 70 && marks <= 80) {
    System.out.println("fair");
} else if (marks > 60 && marks <= 70) {
    System.out.println("meets expectations");
} else {
    System.out.println("below par");
}
}
```

T && F = F

T && T = T

marks = 90

marks = 60

```
    int marks = scn.nextInt();
    if (marks > 90) {
        System.out.println("excellent");
    } else if (marks > 80) {
        System.out.println("good");
    } else if (marks > 70) {
        System.out.println("fair");
    } else if (marks > 60) {
        System.out.println("meets expectations");
    } else {
        System.out.println("below par");
    }
}
```

Grace marks

↳ marks = 70

↳ marks < 33 (+4)

↳ marks >= 33 — ×

Comments in Java → To make code more readable

↳ (1) Single line comment → (//)

↳ (2) Multi-line comment → (/* __ */)

```
        Scanner scn = new Scanner(System.in);

        // int marks = scn.nextInt(); -> Single line Comment
        // condition for grading
        /* if (marks > 90) {
            System.out.println("excellent");
        } else if (marks > 80) {
            System.out.println("good");
        } else if (marks > 70) {
            System.out.println("fair");
        } else if (marks > 60) {
            System.out.println("meets expectations");
        } else {
            System.out.println("below par");
        }*/ => Multi line comment
    }
```

ch = 'a' →

a = 'A' →

ch = '1' →

else → None

'a' - 'z' → Small case

'A' - 'Z' → Capital case

'0' - '9' → digits

None

ch = '#1' → None

ch = 'd' → A

ch >= 'a' && ch <= 'z'

↳ Small case

ASCII

Behind the scence

a b - - - -

↓ ↓

97 98

1007 = 97 && 1004 = 122

ch - 'a' : 3 + 97

Homework - Q.2

G	A	B	C	D	E	F	G
	1	1	1	1	1	1	1
	1	2	3	4	5	6	7

$(A+B+C) * (D+E+F+G)$

$(1+2+3) * (4+5+6+7)$

$(6) * (22) = 132$

String and Loops

String → collection of characters

"NavGurukul"

↳ `String str_name = " ";`

↳ User defined string input

NavGurukul Bootcamp.

↳ scn.next() → NavGurukul

↳ scn.nextLine() → NavGurukul Bootcamp

String in Java is **immutable**

String → non primitive datatype

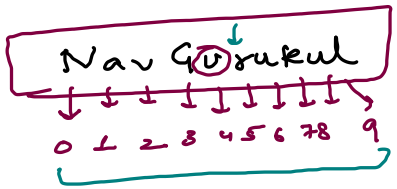
str = "abc"

str.charAt(i) → ith char

str.charAt(5) → 5

string_name.charAt(index)

str.charAt(8) → u



→ indexing

str.charAt(4) → u

str.charAt(9) → l

str.charAt(13) → Error

str.charAt(13)

↳ invalid index → Error

str = "Bootcamp"

Bootcamp
0 1 2 3 4 5 6 7

↳ length() → str.length() → 8

↳ substring

↳ str.substring(starting_index, ending_index)

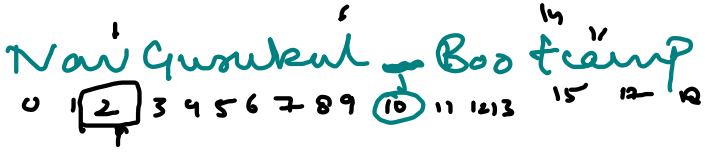
↳ str.substring(starting index);

Nav Gurukul_Bootcamp → 14

str.substring(2) → otcamp

str.substring(3, 7) → tcam

(starting_index, ending_index)



2.5

(2, 9)

str. substring (2, 10) → v Gurukul

Substring → Case - 01

↳ when string is empty means length = 0 (str = "")

→ M/a

Str. substring(0, 0); → no output

2) str. substring(0, 1); → error / index outy range.

3) str. substring(0); → str.substring(0,0) → no output

4) str. substring(-1) → out of range

5) str. substring(-5); → out of rng

6) str. substring(1, 5) → index out of range

7) str. substring(5, 1) → index out of range

8) str. substring(1, 0) → index out of range.

9) str. substring( ) → syntax error

10) str. substring(-1, -5)
↳ index out of range

11) str. substring(0, -2) → index out of range

12) str. substring(-2, 0) → index out of range.

13) str. substring(1) → index out of range

14) str. substring(-2, 5) → index out of range.

15) str. substring(1, 1) → index out of range.

16) str. substring(, 6) → syntax error

17) str. substring(6, 0) → index 0 out of range.

18) str. substring(0, -1) → index out of range.

19) str. substring(5,) → syntax error

20) str. substring(-1, 0) → index out of range.

Obs:- 1) no concept of negative index exist in Java Programming language

Obs:- 2) str. substring(starting-index, ending-index)

↳ last position goes to ending-index.

Obs-3 → starting index cannot be greater than ending index

Obs-4 → when starting index and ending index are same → output → no output.

↳ only when starting and ending index is in range of string length.

Ex → str = abcd → possible values of st, ed → [0, 1, 2, 3, 4]

Case: 02

when length of string is exactly 1

0, 2 → (0, 1)

a → X

str = "a"

0

1. str.substring(0,0) → no output → blank / empty string
2. str.substring(0) → "a"
3. str.substring(1) → no output → blank / empty string
4. str.substring(1,0) → error
5. str.substring(0,1) → "a"
6. str.substring(0,2) → error
7. str.substring(2,0) → error
8. str.substring(1,1) → no output → blank / empty string

Case 03: →

When length of string greater than 1

str = "abcd" → length = (4) → [0, 1, 2, 3]

1. str.substring(0,0) → no output
2. str.substring(0) → abcd
3. str.substring(1) → bcd
4. str.substring(1,0) → error
5. str.substring(0,1) → a
6. str.substring(0,2) → ab
7. str.substring(2,0) → error
8. str.substring(1,1) → no output
9. str.substring(2,2) → no output
10. str.substring(4,4) → no output

↳ when starting index and ending index equal to the length of string

11. str.substring(3,4) → d
12. str.substring(3,3) → no output
13. str.substring(5,5) → error

↳ when starting index and ending index is greater than string length.