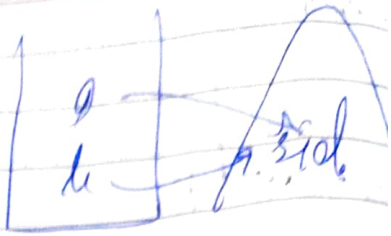
Strings

String name = "Siddharth"

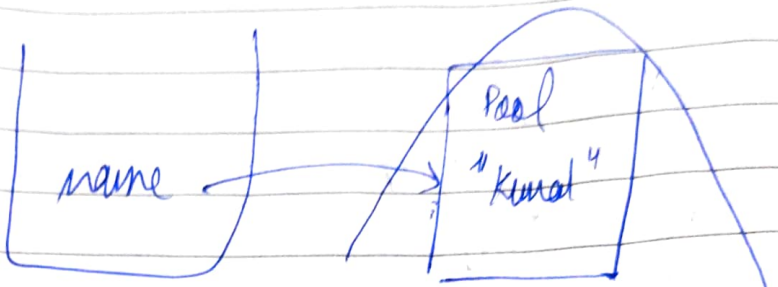
String → Datatype.
name → Ref. var.
"Siddharth" → Object.

- String is a class

String a = "Sid"
String b = "Sid"



String Pool



Pool is a special memory structure in the heap.

Immutability

Strings are immutable in Java. Objects can't be changed. A new object has to be created.

Comparision of Strings →

① == method

↳ Comparator

Returns true if ref variables are pointing to the same object.

How to create diff objects of same value →

- Using the 'new' keyword.

String a = new String ("Sid")
String b = new String ("Sid")

a == b // false.

- When you only need to check value, use .equals method.

- In strings → Although its an array, we can't use
name [0] // Wrong
name.charAt(0) // Right.

- Println calls valueOf() method which calls toString method.

Sout (new int[]{1, 3, 5});
→ o/p [I@3ca5ee0.

← prints actual string representation of the object (Using Hashcode @ & some random String).

— this is not beneficial to us.
— So we use the toString method available in Arrays.

Sout (Arrays.toString (new int[]{1, 3, 5}));
→ o/p [1, 3, 5].

- Integer num = new Integer (56)
Sout (num.toString());
Or Sout (num);

Using a wrapper class which has a method toString.

float a = 453.1274f;

cout.printf("Formatted no. is %.2f", a);

Placeholder Till 2 decimal place

→ o/p 453.12

if a = 453.1274f;
→ o/p 453.13 // rounding off.

Placeholders →

%c.	Character
%d	Decimal No.
%e	Exponential – floating point No
%f	Floating point No.
%i	Integer
%o	Octal No.
%s	String
%u	Unsigned decimal (integer) no.
%x.	Hexadecimal No.
%t	Date / time
%n	New line.

→ cout ('a' + 'b'); // 195
cout ("a" + "b"); // ale

↳ Operator overloading of '+'

→ Sout ('a' + 3); // 100.

→ Sout ("a" + 3); // a2

- when an integer is contacted to string, the int is converted into its wrapper class Integer which calls toString function.

- The operator '+' is only applied to primitives & when one of the operands is a String.

→ Sout (new Integer 156) + new ArrayList<>());

→ error.

→ Sout (new Integer (56) + "n" + new ArrayList<>());

→ 56[]

☆ StringBuilder

Let String = ""

① s = s + "a" = "a"

② s = "a" + "b" = "ab"

"ab" + "c" = "abc".

- s is created by a new object of String created every time (since strings are immutable)

- we need a datatype that'll allow us to modify it.

- StringBuilder.

to do StringBuilder builder = new StringBuilder();

one.