Mutable Strings

Immutable
new String("abc")
^abc^

① String Buffer
② String Builder

① String Buffer:

StringBuffer   str = new StringBuffer("abc")
                                        ↑ → Type

str = "abc"

Drawback:
↓
Not allowed

str 1000

2000 ("abc")

"abc" "def"
"def"
(abc)

SCP
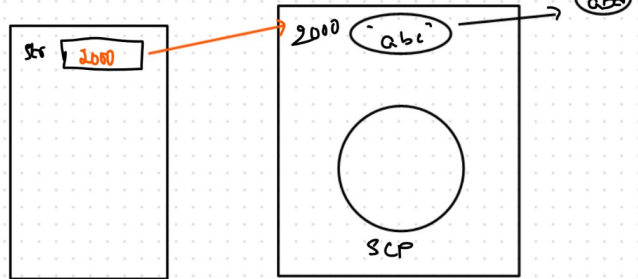
① Entire String replace
② Selected index replacement

① Entire String

str = "abc"
     ↑ ↑

= str.replace(0, str.length(), "defg")
              ↑   ↑   ↑
            Start   end   new String value
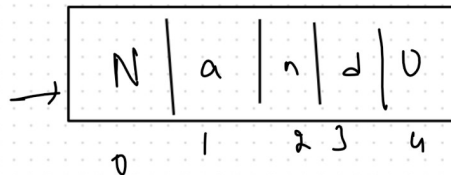   < exclude
                       5

str = ""

str.replace(0, 0, "abc");

② Selected index replacement

str = "Nandu" → N @ndu   "a"   "@"
      ↑   str.indexOf("a"), str.indexOf("a"), "@")
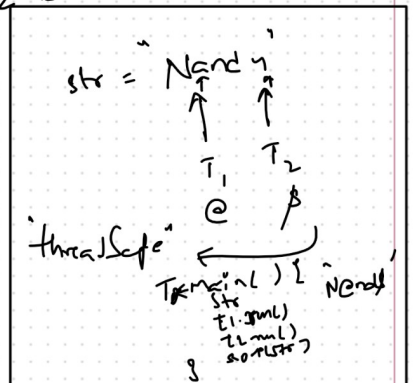
str = str.replace(str.indexOf("a"), str.indexOf("a"), "@")

N	a	n	d	U
0	1	2	3	4

This

C.le(1) (or middle)
↓
N @ n d y   0 ✗
   ↑   N@ndy
  1,1   0 1 2 3 4 5
 (1,2)

C.le(2) (end)   Nandu   ✗
Nand @
0 1 2 3 4
↑
str.replace(4, 4, "@")

inclusive → include
exclusive → exclude except
            that index

I   E
①   ②
↑
②   0,1 ✗
③
↑   0 1
       0 12 ✗

str = "Nandy"
       ↑   ↑
       T₁   T₂
       T₁   T₂
            @
"threadSafe" ←
              Terminal) ₹ "N@ndy"
                      str
                      t1.join()
                      t2.join()
                      ₹ s.o.p(str)

Note: StringBuffer is "threadSafe" & String Builder is not

String formatting

without format :

→ "Hi, My name " + str + " I am from " + location

with formatting :

name = "Sai"
location = "Blr"

String result = String.format(" Hi, my name is %s , I am from %s ", name, location);
S.o.P(result);

%s → ?

Format Specifiers

Specifier	Type
%d	integer
%f	float / double
%s	string
%c	char
%b	boolean
%e	integer (scientific notation)
%%	literal (pacent sign %)

50%

%

e⁻¹⁰ → e^{-10}

Example:

String name = "Sai"
int age = 24

→ (Name : Sai , Age : 24)

String result = String.format(" Name : %s , Age : %d ", name, age);
S.o.P(String.format(" "))

S.o.P(result)

② Width and Precision

Precision

double price = 123.4567

%.f×

String.format("price : %.2f ", price);

→ 123.46
→ 00123.45

Note: set the number of digits after the decimal point   %._nf

Width: 123.456754 →

%.8.2f

%.08.2f

```
      1  2  3  .  4  6
0  1  2  3  4  5  6  7
      123.46
```

Note: Specifies the minimum number of characters to be printed

above en = .2f limits 2 decimal place

%.8.2f → make it with 8 character

---

④ **formatting flags**

Flag	description
−	left-justify the output
+	Add a +sign for a given positive variable
0	pads number with zeros instead of spaces

egt

int number = 42   +42   '+42'

                                                42

String format ("%.+d", number) → +42   00042
                                                 5
String format ("%.05d", number)

                                        25ch

Name	Description
Sa	
Nandu	
Nandgoh Pla	

String format ("%.-25s", "Name")

name _____ Description

String format ("%.-25s", "%s")
Sa _____

Renaming on   ┌──────────────┐
                     │ + dd mm yyyy │   ┌──────────┐
DTe & time   │ → yyyy mmdd │   │ Date & Time │ ←
                     │ → mm dd yyyy │   └──────────┘
Formatting   │ → dd - mm-yyyy│
                     └──────────────┘   Htt:mm:ss   dd - mm-yy

---

Escape sequence)   %.+Y
                     %.+m   String format ("_")
                                                ⇓
┌──────────────┐   System.out.printf ("_")
│ \t   \n   \a │
└──────────────┘   - - - -