Strings - collection of characters.
Ex:- string s = "Algorithm";

Characters -

a) Alphabet
   ↳ a - z (lowercase)
   ↳ A - Z (uppercase)

b) Special Characters
   ↳ @ # $? ! % ^ ~ * &

c) Numbers
   ↳ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

Ex:- char ch = '1';

for any character there is a integer value associated with it and it is called as ASCII values.

char ch = 'B'; // maps to 66 in ASCII value.

In total, there are a total of **256** characters in ASCII.

ASCII values for characters

'A'	→	65
'Z'	→	90
'a'	→	97
'0'	→	48

Character rules

1. can do any mathematical operation on character and the answer will be integer

    print ('A' + 'B') -> will be 65 + 66 = 131

2. Typecasting

(a) char to int will be implicit.

```
int x = 'C';    // 67
```

(b) int to char (complicated)

```
-> char ch = 66 ;   // 'B'
-> char ch4 = 'A';
   ch4 = ch4 + 3;   // error
   print (ch4);
```

$\rightarrow$ do explicit conversion

```
char ch = (char) 66;
```

In few cases, it will be implicit and in few cases, it is explicit.

Always do an explicit conversion/typecast.

implicit conversions	explicit conversion
long x = 10;	char ch = (char) 67;
int x = 'A';	

Taking input:

```
Scanner scn = new Scanner (System.in);
char ch = scn.nextLine().charAt(0);
```

Uppercase & lowercase

Ex:- ALGORITHMS → algorithms

$$'A' : 65 \xrightarrow{+32} 'a' : 97$$

$$'B' : 66 \xrightarrow{+32} 'b' : 98$$

ASCII difference between uppercase & lowercase is +32.

uppercase to lowercase → +32

lowercase to uppercase → -32

NOTE: a. whenever you want to change characters, think in terms of ASCII,

b. Range condition from A to Z would be
ch[i] >= 65 && ch[i] <= 90 given ch is an array.

c. Range condition from a to z would be
ch[i] >= 97 && ch[i] <= 122.

Reverse a string

```
public static String reverseString(String str){
    char[] ch = str.toCharArray();   → extra memory which is not an input array
    int sp= 0;
    int ep= str.length() -1; (OR) ch.length -1;
    while(sp < ep){
        char temp = ch[sp];
        ch[sp]= ch[ep];
        ch[ep] = temp;

        sp++;
        ep--;
    }
    return "".valueOf(ch);
}
```

∴ T.C - O(n)
S.C - O(n)

NOTE - Do not concatenate in Strings since the time complexity is O(N). Since strings are immutable, a new string object would be created, copied into it. For copying there is a loop that's running internally.