Strings & Characters

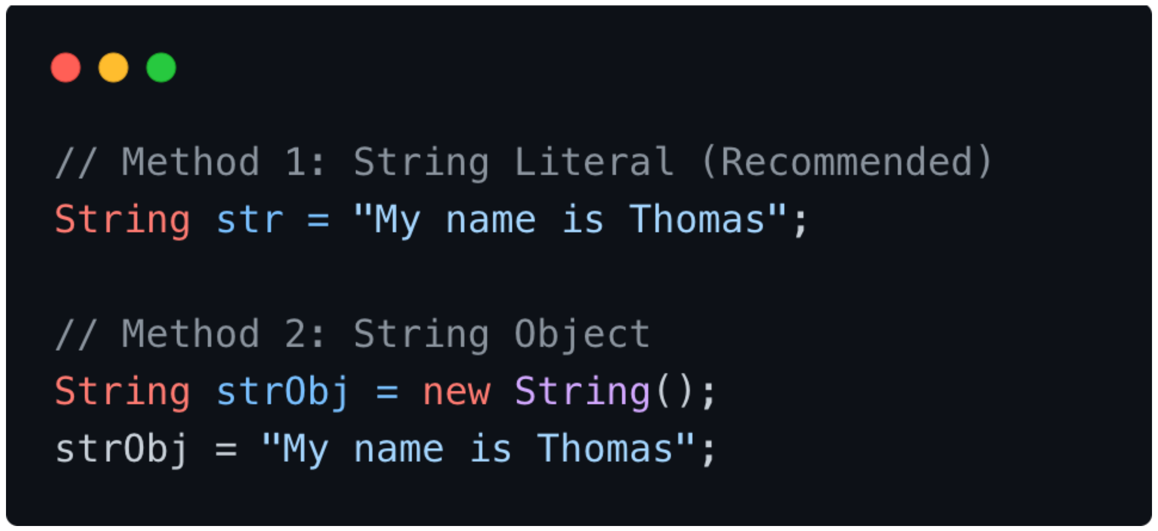
ENG23 2032

Object–Oriented Technology

Dr. Nuntawut Kaoungku
Assistant Professor of Computer Engineering

Introduction to Strings

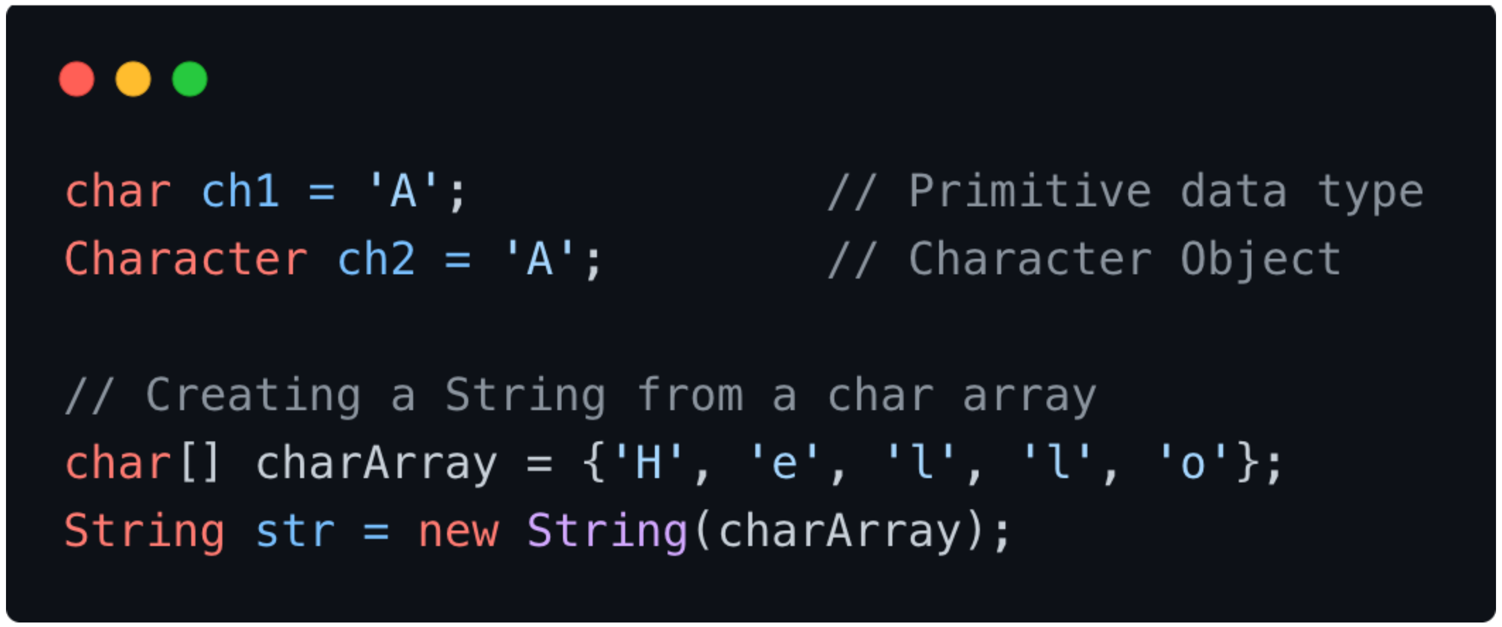
- **Definition:** Strings are used for storing text.
- **Structure:** A String variable contains a collection of characters surrounded by double quotes.
- **Immutability:** Strings are constant; their values cannot be changed after they are created.



```java
// Method 1: String Literal (Recommended)
String str = "My name is Thomas";

// Method 2: String Object
String strObj = new String();
strObj = "My name is Thomas";
```

The Character Class

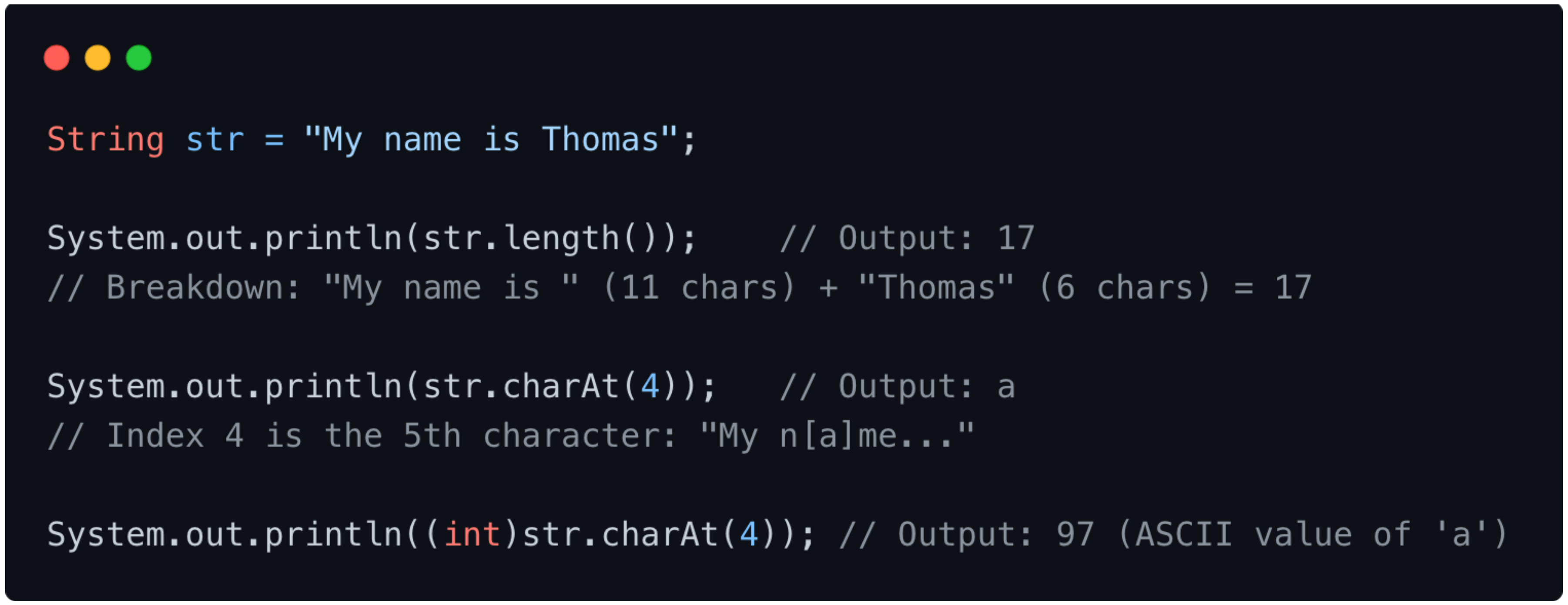
- **Primitive (char)**: Used to store a single character. Must be surrounded by single quotes (e.g., 'A').
- **Wrapper (Character)**: The Character class wraps a value of the primitive type char in an object.



```java
char ch1 = 'A';                // Primitive data type
Character ch2 = 'A';           // Character Object

// Creating a String from a char array
char[] charArray = {'H', 'e', 'l', 'l', 'o'};
String str = new String(charArray);
```

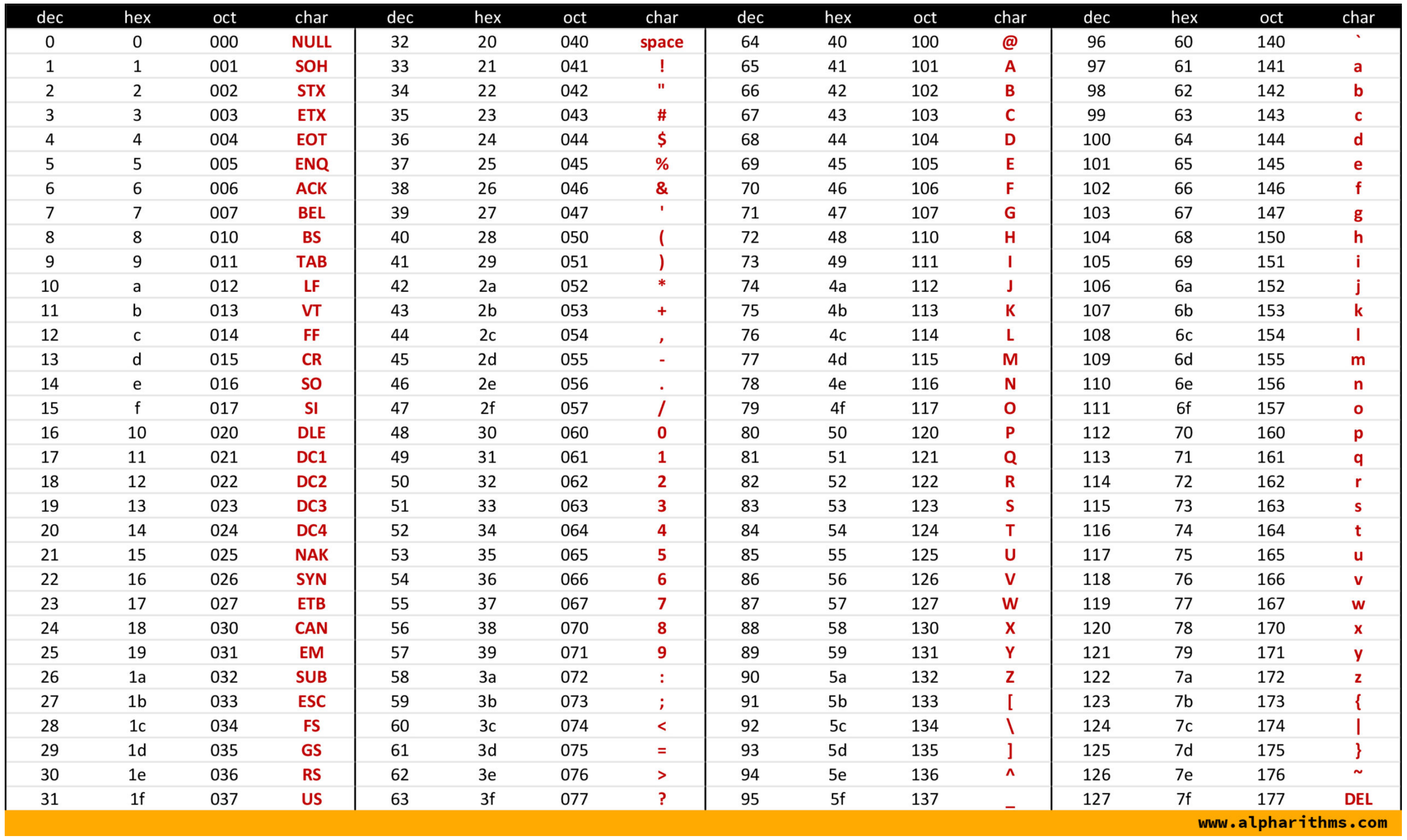
Basic String Methods

- **length( ):** Returns the length of the string (number of characters).
- **charAt(int index)**: Returns the char value at the specified index.
  - Index ranges from 0 to length() - 1.



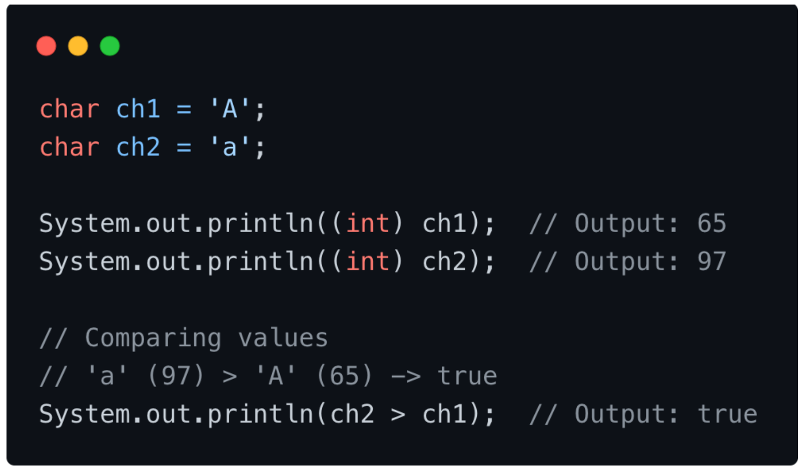
```java
String str = "My name is Thomas";

System.out.println(str.length());    // Output: 17
// Breakdown: "My name is " (11 chars) + "Thomas" (6 chars) = 17

System.out.println(str.charAt(4));   // Output: a
// Index 4 is the 5th character: "My n[a]me..."

System.out.println((int)str.charAt(4)); // Output: 97 (ASCII value of 'a')
```

ASCII Code



dec	hex	oct	char	dec	hex	oct	char	dec	hex	oct	char	dec	hex	oct	char
0	0	000	NULL	32	20	040	space	64	40	100	@	96	60	140	`
1	1	001	SOH	33	21	041	!	65	41	101	A	97	61	141	a
2	2	002	STX	34	22	042	"	66	42	102	B	98	62	142	b
3	3	003	ETX	35	23	043	#	67	43	103	C	99	63	143	c
4	4	004	EOT	36	24	044	$	68	44	104	D	100	64	144	d
5	5	005	ENQ	37	25	045	%	69	45	105	E	101	65	145	e
6	6	006	ACK	38	26	046	&	70	46	106	F	102	66	146	f
7	7	007	BEL	39	27	047	'	71	47	107	G	103	67	147	g
8	8	010	BS	40	28	050	(	72	48	110	H	104	68	150	h
9	9	011	TAB	41	29	051	)	73	49	111	I	105	69	151	i
10	a	012	LF	42	2a	052	*	74	4a	112	J	106	6a	152	j
11	b	013	VT	43	2b	053	+	75	4b	113	K	107	6b	153	k
12	c	014	FF	44	2c	054	,	76	4c	114	L	108	6c	154	l
13	d	015	CR	45	2d	055	-	77	4d	115	M	109	6d	155	m
14	e	016	SO	46	2e	056	.	78	4e	116	N	110	6e	156	n
15	f	017	SI	47	2f	057	/	79	4f	117	O	111	6f	157	o
16	10	020	DLE	48	30	060	0	80	50	120	P	112	70	160	p
17	11	021	DC1	49	31	061	1	81	51	121	Q	113	71	161	q
18	12	022	DC2	50	32	062	2	82	52	122	R	114	72	162	r
19	13	023	DC3	51	33	063	3	83	53	123	S	115	73	163	s
20	14	024	DC4	52	34	064	4	84	54	124	T	116	74	164	t
21	15	025	NAK	53	35	065	5	85	55	125	U	117	75	165	u
22	16	026	SYN	54	36	066	6	86	56	126	V	118	76	166	v
23	17	027	ETB	55	37	067	7	87	57	127	W	119	77	167	w
24	18	030	CAN	56	38	070	8	88	58	130	X	120	78	170	x
25	19	031	EM	57	39	071	9	89	59	131	Y	121	79	171	y
26	1a	032	SUB	58	3a	072	:	90	5a	132	Z	122	7a	172	z
27	1b	033	ESC	59	3b	073	;	91	5b	133	[	123	7b	173	{
28	1c	034	FS	60	3c	074	<	92	5c	134	\	124	7c	174	|
29	1d	035	GS	61	3d	075	=	93	5d	135	]	125	7d	175	}
30	1e	036	RS	62	3e	076	>	94	5e	136	^	126	7e	176	~
31	1f	037	US	63	3f	077	?	95	5f	137	_	127	7f	177	DEL

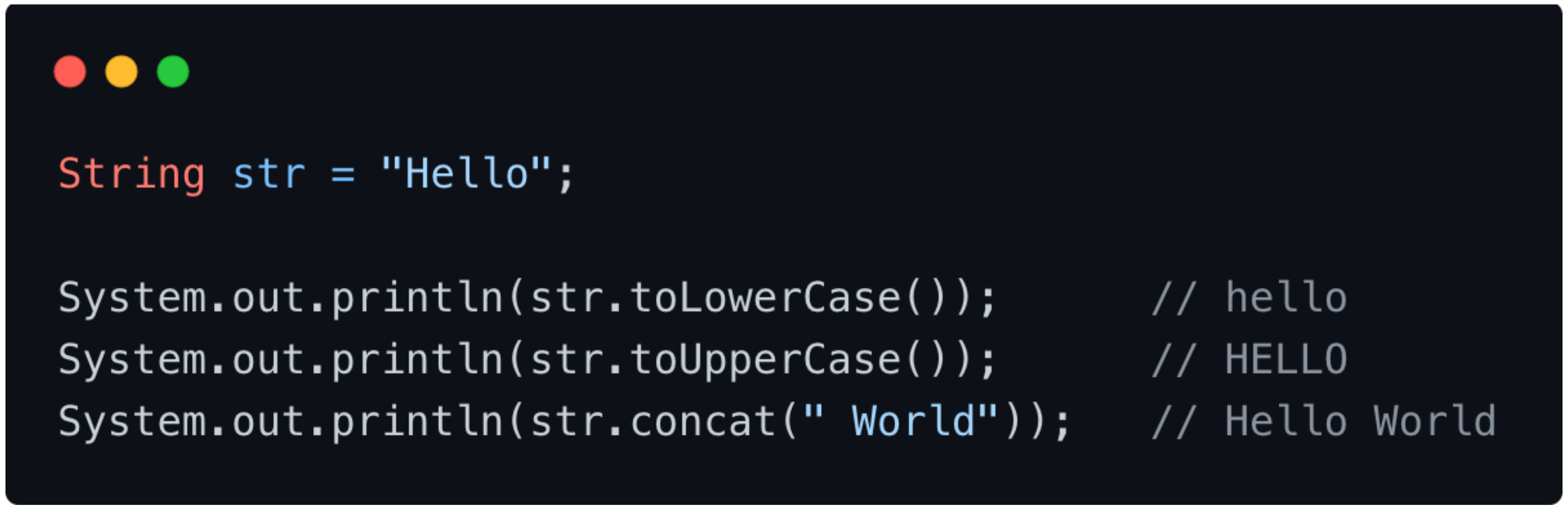
www.alpharithms.com



```java
char ch1 = 'A';
char ch2 = 'a';

System.out.println((int) ch1);  // Output: 65
System.out.println((int) ch2);  // Output: 97

// Comparing values
// 'a' (97) > 'A' (65) -> true
System.out.println(ch2 > ch1);  // Output: true
```

Modifying Strings

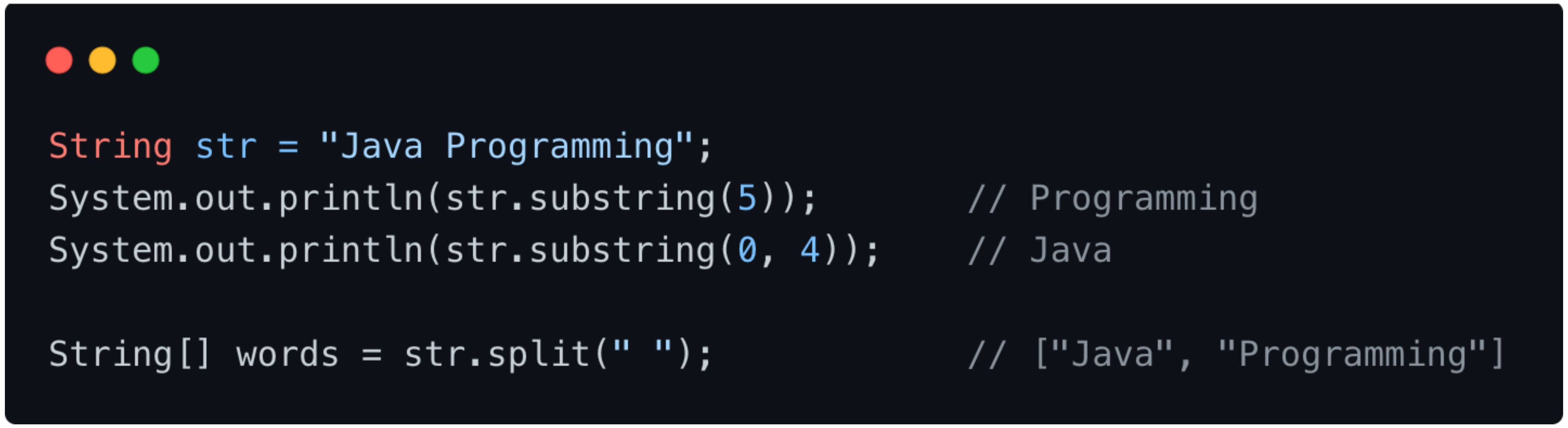
- **toLowerCase():** Converts all characters to lower case.
- **toUpperCase():** Converts all characters to upper case.
- **concat(String str):** Concatenates (joins) the specified string to the end.



```
String str = "Hello";

System.out.println(str.toLowerCase());     // hello
System.out.println(str.toUpperCase());     // HELLO
System.out.println(str.concat(" World"));  // Hello World
```

Substrings and Splitting

- **substring(int beginIndex)**: Returns a substring starting from the specified index to the end.
- **substring(int begin, int end)**: Returns a substring starting from begin up to (but not including) end.
- **split(String regex)**: Splits the string around matches of the given regular expression.

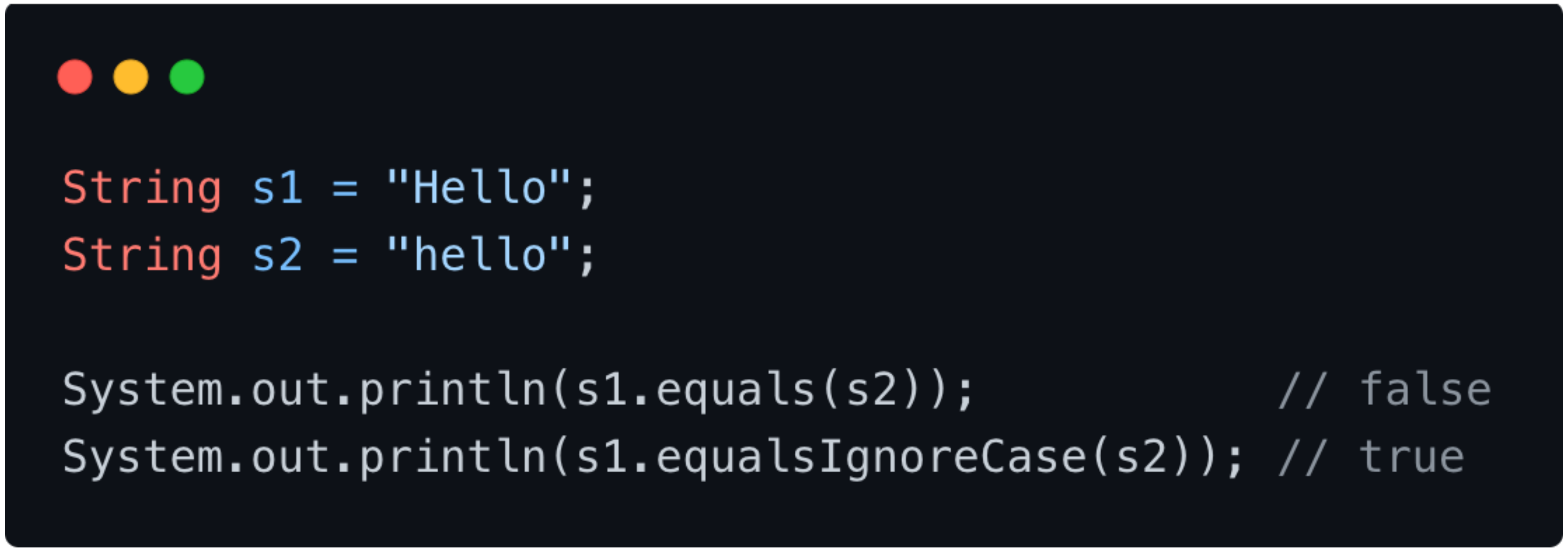


```java
String str = "Java Programming";
System.out.println(str.substring(5));        // Programming
System.out.println(str.substring(0, 4));     // Java

String[] words = str.split(" ");             // ["Java", "Programming"]
```

String Comparison (Equality)

- **equals(Object anObject):** Compares the string to the specified object. Returns true if they are exactly equal.

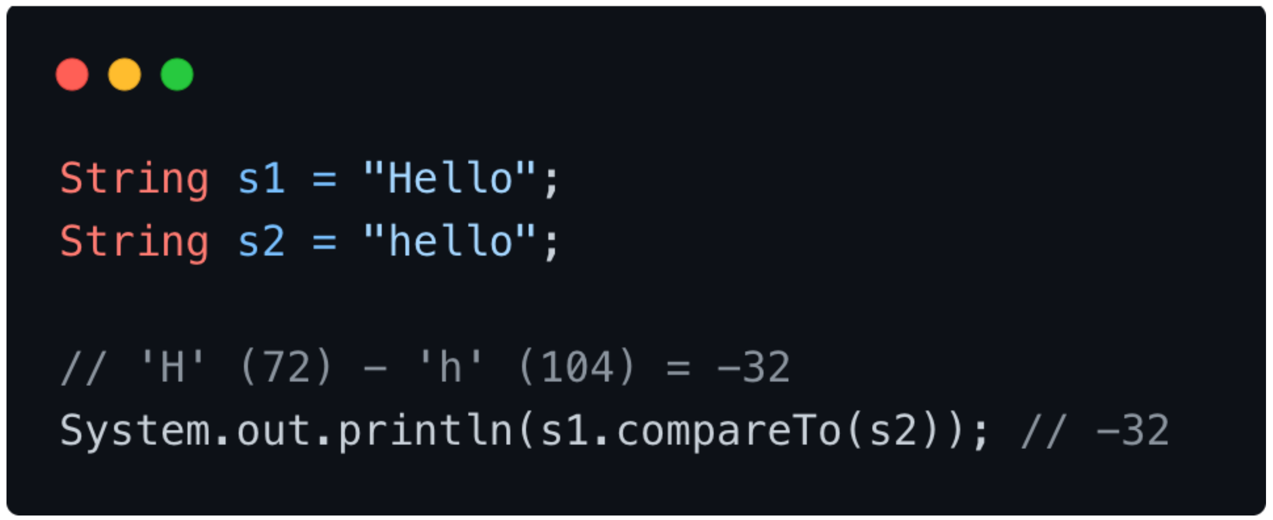
- **equalsIgnoreCase(String anotherString):** Compares two strings, ignoring case considerations.



```
String s1 = "Hello";
String s2 = "hello";

System.out.println(s1.equals(s2));           // false
System.out.println(s1.equalsIgnoreCase(s2)); // true
```

Lexicographical Comparison

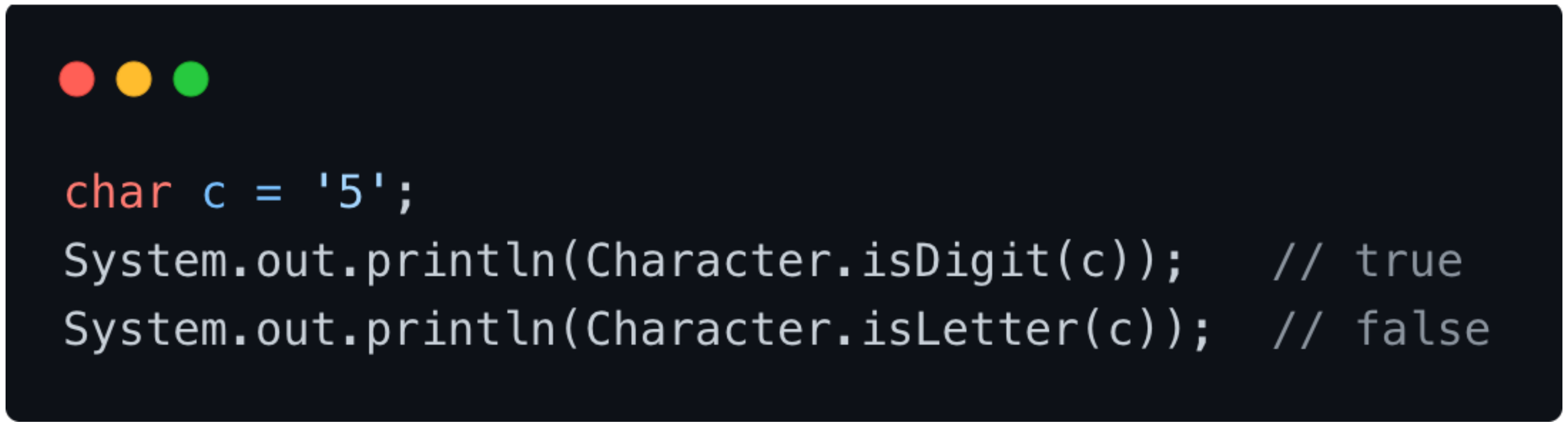
- **compareTo(String anotherString)**: Compares two strings lexicographically (based on Unicode value).
    - ***Returns 0***: if strings are equal.
    - ***Returns Negative***: if current string < anotherString.
    - ***Returns Positive***: if current string > anotherString.



```java
String s1 = "Hello";
String s2 = "hello";

// 'H' (72) - 'h' (104) = -32
System.out.println(s1.compareTo(s2)); // -32
```

Character Helper Methods

- **The Character class provides static methods to test char values:**
  - ***isDigit(char ch)***: Is it a number (0-9)?
  - ***isLetter(char ch)***: Is it an alphabet letter?
  - ***isSpaceChar(char ch)***: Is it a space?
  - ***isLowerCase(char ch) / isUpperCase(char ch)***: Check case.



```java
char c = '5';
System.out.println(Character.isDigit(c));   // true
System.out.println(Character.isLetter(c));  // false
```

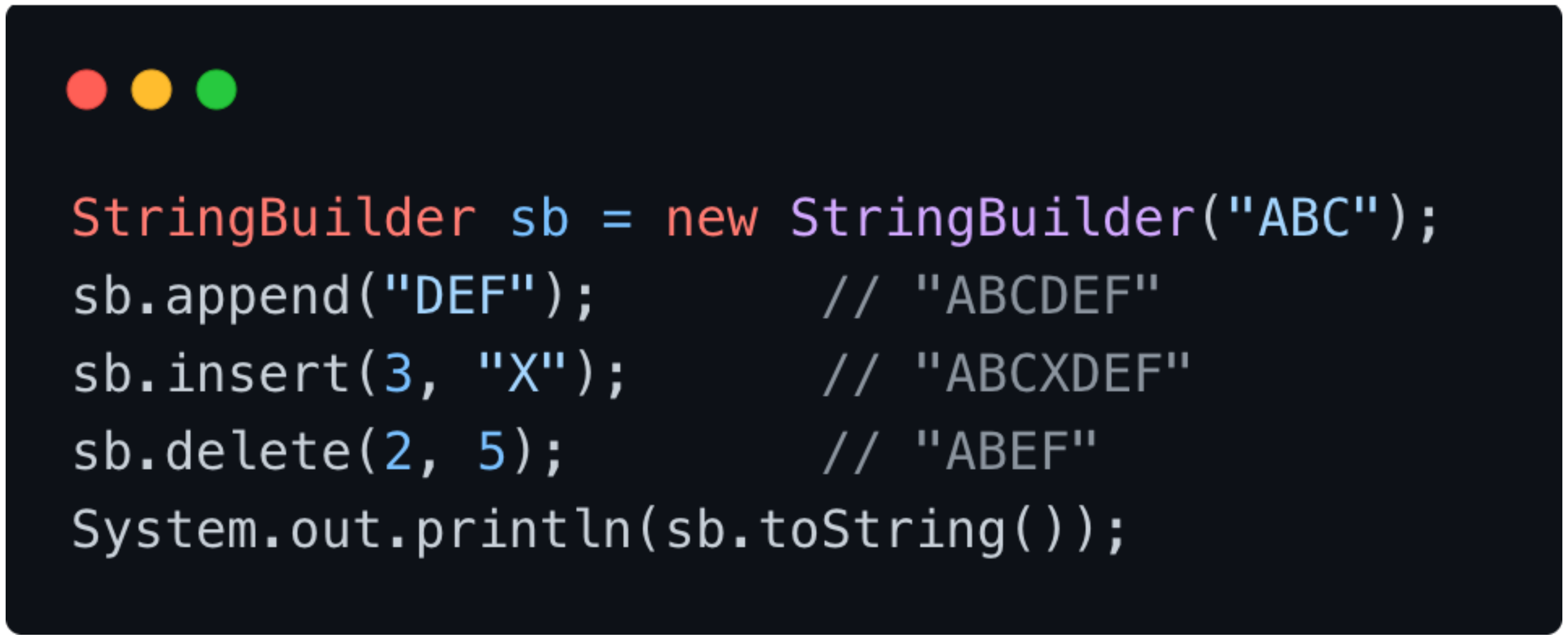
Mutable Strings (Buffer & Builder)

Java provides classes for strings that can be modified:

1. ***StringBuffer:***
    - Mutable.
    - Thread-safe (Synchronized).
    - Slower.
2. ***StringBuilder:***
    - Mutable.
    - Not Thread-safe (Not synchronized).
    - Faster (Recommended).

StringBuilder Methods

- Common methods for both StringBuffer and StringBuilder:
    - *append(), insert(), delete(), toString().*

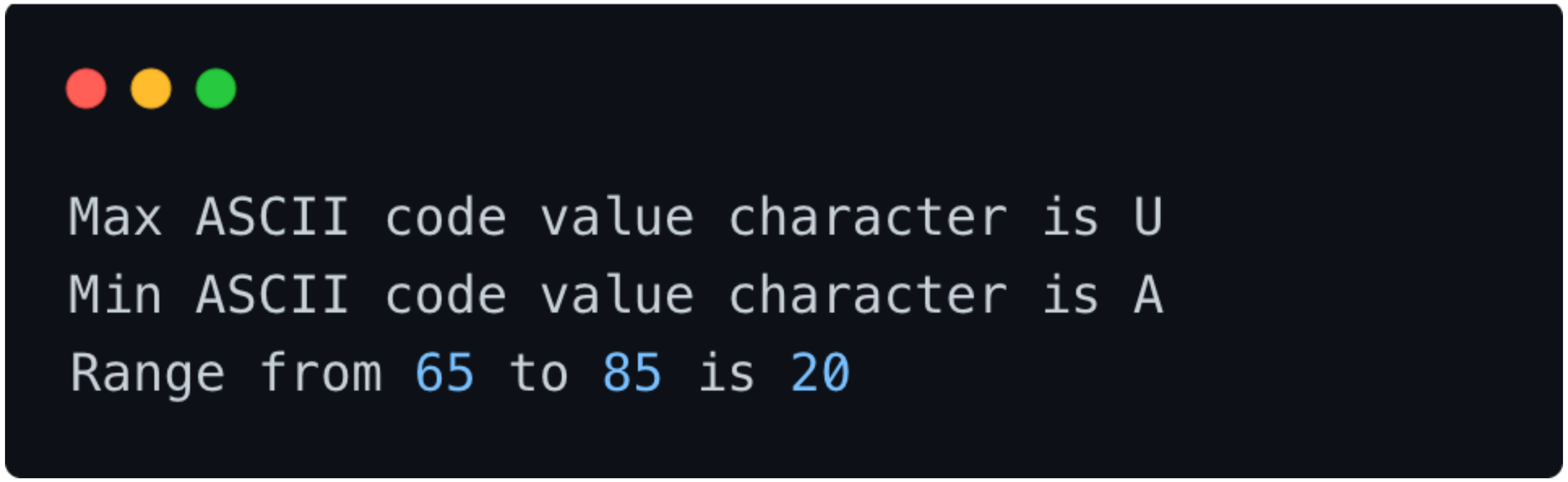


```java
StringBuilder sb = new StringBuilder("ABC");
sb.append("DEF");         // "ABCDEF"
sb.insert(3, "X");        // "ABCXDEF"
sb.delete(2, 5);          // "ABEF"
System.out.println(sb.toString());
```

Exercise 1

- **Task:** *Write a Java program to find the characters with the highest and lowest ASCII code values from the string "AEIOU".*

- **Expected Output:**

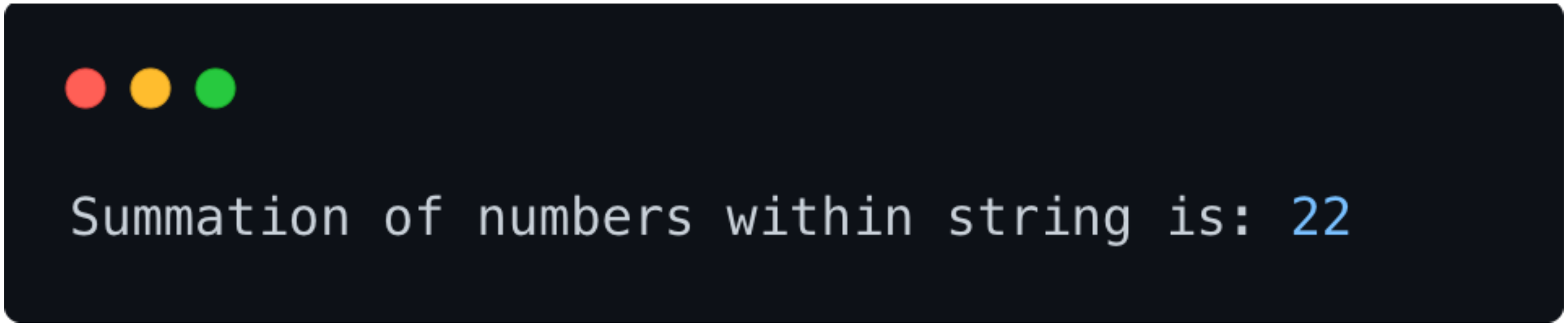


```
Max ASCII code value character is U
Min ASCII code value character is A
Range from 65 to 85 is 20
```

Exercise 2

- **Task:** *Write a Java program to find a summation of numbers within the string "asdfi23089f%".*

- **Expected Output:**

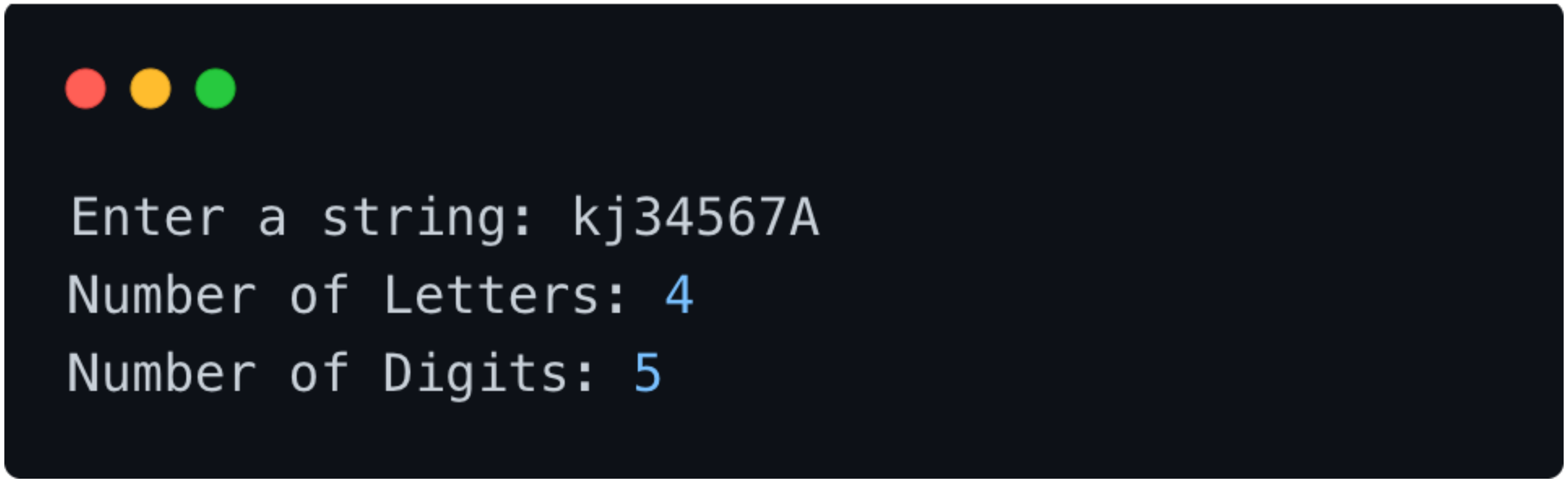


```
Summation of numbers within string is: 22
```

Exercise 3

- **Task:** *Write a Java program to count and display the number of letters and digits of the input string.*

- **Expected Output:**



```
Enter a string: kj34567A
Number of Letters: 4
Number of Digits: 5
```