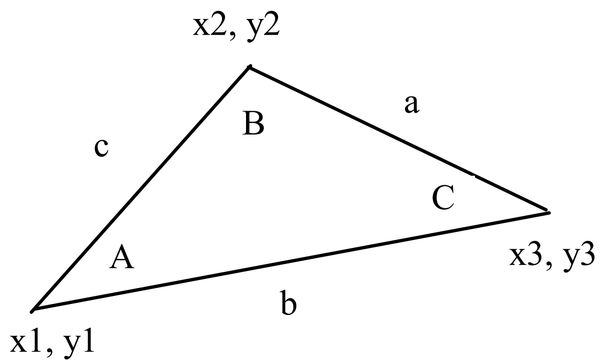
Chapter 4 Mathematical Functions, Strings, and Objects

What to be covered next

- Python Built-in Functions
- Strings and Characters
- Unicode and ASCII Code
- Escape Sequences for Special Characters
- Printing without the Newline
- The *str* Function
- Reading Strings from the Console

Problem: Compute Angles

Given three points of a triangle, you can compute the angles using the following formula:



```
A = acos((a * a - b * b - c * c) / (-2 * b * c))
B = acos((b * b - a * a - c * c) / (-2 * a * c))
C = acos((c * c - b * b - a * a) / (-2 * a * b))
```

ComputeAngles

https://liangcpp.pearsoncmg.com/pyhtml/ComputeAngles.html

Python Built-in Functions

Function	Description	Example
abs(x)	Returns the absolute value for x.	abs(-2) is 2
max(x1, x2, ...)	Returns the largest among x1, x2, ...	max(1, 5, 2) is 5
min(x1, x2, ...)	Returns the smallest among x1, x2, ...	min(1, 5, 2) is 1
pow(a, b)	Returns a^b. Same as a ** b.	pow(2, 3) is 8
round(x)	Returns an integer nearest to x. If x is equally close to two integers, the even one is returned.	round(5.4) is 5 round(5.5) is 6 round(4.5) is 4
round(x, n)	Returns the float value rounded to n digits after the decimal point.	round(5.466, 2) is 5.47 round(5.463, 2) is 5.46

Built-in Functions

```
>>> max(2, 3, 4) # Returns a maximum number
4
>>> min(2, 3, 4) # Returns a minimu number
2
>>> round(4.51) # Rounds to its nearest integer
4
>>> round(4.4) # Rounds to its nearest integer
3
>>> abs(-3) # Returns the absolute value
3
>>> pow(2, 3) # Same as 2 ** 3
8
```

The math Functions

Function	Description	Example
fabs(x)	Returns the absolute value of the argument.	fabs(-2) is 2
ceil(x)	Rounds x up to its nearest integer and returns this integer.	ceil(2.1) is 3 ceil(-2.1) is -2
floor(x)	Rounds x down to its nearest integer and returns this integer.	floor(2.1) is 2 floor(-2.1) is -3
exp(x)	Returns the exponential function of x (e ** x).	exp(1) is 2.71828
log(x)	Returns the natural logarithm of x.	log(2.71828) is 1.0
log(x, base)	Returns the logarithm of x for the specified base.	log10(10, 10) is 1
sqrt(x)	Returns the square root of x.	sqrt(4.0) is 2
sin(x)	Returns the sine of x. x represents an angle in radians.	sin(3.14159 / 2) is 1 sin(3.14159) is 0
asin(x)	Returns the angle in radians for the inverse of sine.	asin(1.0) is 1.57 asin(0.5) is 0.523599
cos(x)	Returns the cosine of x. x represents an angle in radians.	cos(3.14159 / 2) is 0 cos(3.14159) is -1
acos(x)	Returns the angle in radians for the inverse of cosine.	acos(1.0) is 0 acos(0.5) is 1.0472
tan(x)	Returns the tangent of x. x represents an angle in radians.	tan(3.14159 / 4) is 1 tan(0.0) is 0
fmod(x, y)	Returns the remainder of x/y as double.	fmod(2.4, 1.3) is 1.1
degrees(x)	Converts angle x from radians to degrees	degrees(1.57) is 90
radians(x)	Converts angle x from degrees to radians	radians(90) is 1.57

```
import math # import Math module to use the math functions

# Test algebraic functions
print("exp(1.0) =", math.exp(1))
print("log(3.78) =", math.log(math.e))
print("log10(10, 10) =", math.log(10, 10))
print("sqrt(4.0) =", math.sqrt(4.0))

# Test trigonometric functions
print("sin(PI / 2) =", math.sin(math.pi / 2))
print("cos(PI / 2) =", math.cos(math.pi / 2))
print("tan(PI / 2) =", math.tan(math.pi / 2))
print("degrees(1.57) =", math.degrees(1.57))
print("radians(90) =", math.radians(90))
```

Strings and Characters

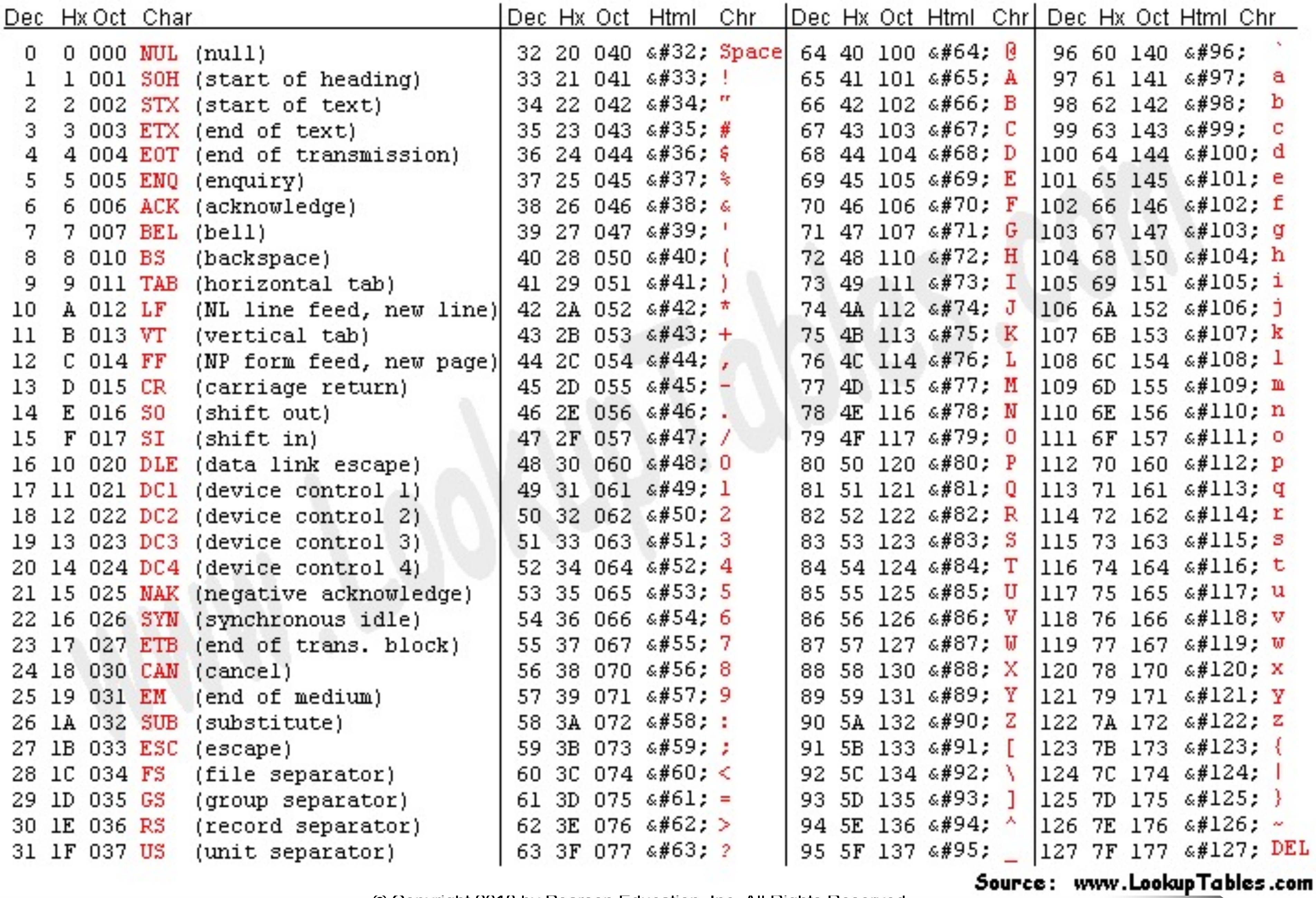
A string is a sequence of characters. *String* literals can be enclosed in matching *single quotes* (') or *double quotes* ("). Python does not have a data type for characters. A single-character string represents a character.

```
letter = 'A' # Same as letter = "A"
numChar = '4' # Same as numChar = "4"
message = "Good morning"
# Same as message = 'Good morning'
```

Unicode and ASCII Code

Python characters use *Unicode*, a 16-bit encoding scheme. Unicode is an encoding scheme for representing international characters. ASCII is a small subset of Unicode.

ASCII Table and Description



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

Source: www.LookupTables.com

Functions ord and chr

```
>>> ch = 'a'
>>> ord(ch)
97
>>> chr(98)
'b'
```

Escape Sequences for Special Characters

Description	*Escape Sequence*	*Unicode*
Backspace	`\b`	`\u0008`
Tab	`\t`	`\u0009`
Linefeed	`\n`	`\u000A`
Carriage return	`\r`	`\u000D`
Backslash	`\\`	`\u005C`
Single Quote	`\'`	`\u0027`
Double Quote	`\"`	`\u0022`

Printing without the Newline

```
print(item, end = 'anyendingstring')
```

```
print("AAA", end = ' ')
print("BBB", end = '')
print("CCC", end = '***')
print("DDD", end = '***')
```

The str Function

The str function can be used to convert a number into a string. For example,

```
>>> s = str(4.4) # Convert a float to string
>>> s
'4.4'
>>> s = str(3) # Convert an integer to string
>>> s
'3'
>>>
```

The String Concatenation Operator

You can use the + operator add two numbers. The + operator can also be used to concatenate (combine) two strings. Here are some examples:

```
>>> message = "Welcome " + "to " + "Python"
>>> message
'Weclome to Python'
>>> chapterNo = 2
>>> s = "Chapter " + str(chapterNo)
>>> s
'Chapter 2'
>>>
```

Problem: Process a string

- Write a program that prompts the user to enter a string and displays its length and its first character.

```
Enter a string: Programming is fun
The length of string Programming is fun is 18
The first character of string Programming is fun is P
```

What to be covered next

- ✦ Introduction to Objects and Methods
- ✦ Str Object Methods
- ✦ Formatting Numbers and Strings

Introduction to Objects and Methods

In Python, all data—including numbers and strings—are actually objects.

An object is an entity. Each object has an id and a type. Objects of the same kind have the same type. You can use the **id** function and **type** function to get these information for an object.

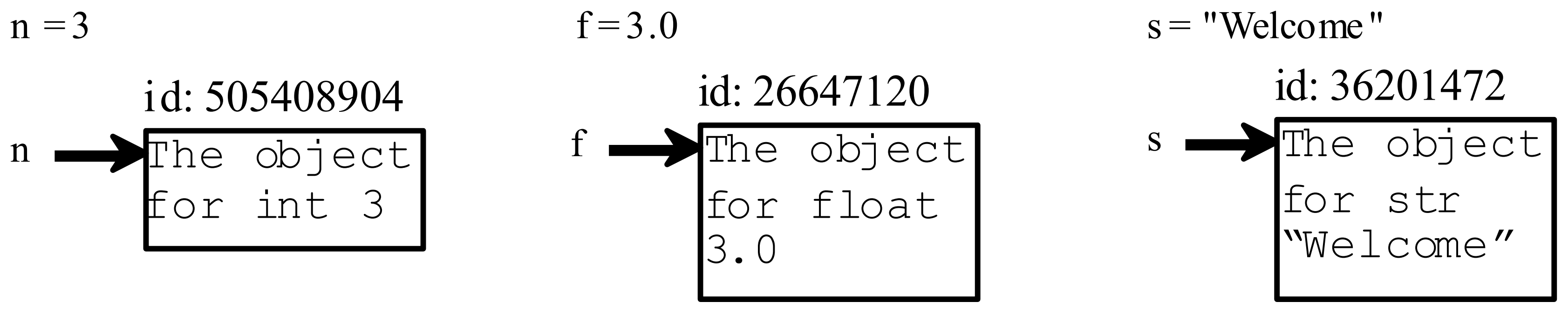
Object Types and Ids

The **id** and **type** functions are rarely used in programming, but they are good pedagogical tools for understanding objects.

```
>>> n = 3  # n is an integer
>>> id(n)
505408904
>>> type(n)
<class ’int’>
>>> f = 4.0  # f is a float
>>> id(f)
26647120
>>> type(f)
<class ’float’>
```

```
>>> s = "Welcome" # s is a string
>>> id(s)
36201472
>>> type(s)
<class ’str’>
```

OOP and str Objects



The **id** and **type** functions are rarely used in programming, but they are good pedagogical tools for understanding objects.

Object vs. Object reference Variable

For n = 3, we say n is an integer variable that holds value 3. Strictly speaking, n is a variable that references an int object for value 3. For simplicity, it is fine to say n is an int variable with value 3.

Methods

You can perform operations on an object. The operations are defined using functions. The functions for the objects are called *methods* in Python. Methods can only be invoked from a specific object. For example, the string type has the methods such as *lower()* and *upper(),* which returns a new string in lowercase and uppercase. Here are the examples to invoke these methods:

str Object Methods

```
>>> s = "Welcome"
>>> s1 = s.lower() # Invoke the lower method
>>> s1
'welcome'
>>> s2 = s.upper() # Invoke the upper method
>>> s2
'WELCOME'
```

Striping beginning and ending Whitespace Characters

Another useful string method is strip(), which can be used to strip the whitespace characters from the both ends of a string.

```
>>> s = "\t Welcome \n"
>>> s1 = s.strip() # Invoke the strip method
>>> s1
'Welcome'
```

Testing Strings

Method	Description
isalnum()	Returns True if all characters in this string are alphanumeric and there is at least one character.
isalpha()	Returns True if all characters in this string are alphabetic and there is at least one character.
isdigit()	Returns True if this string contains only number characters.
isidentifier()	Returns True if this string is a Python identifier.
islower()	Returns True if all characters in this string are lowercase letters and there is at least one character.
isupper()	Returns True if all characters in this string are uppercase letters and there is at least one character.
isspace()	Returns True if this string contains only whitespace characters.

Searching for Substrings

Method	**Description**
endswith(s1)	Returns True if the string ends with the substring s1.
startswith(s1)	Returns True if the string starts with the substring s1.
find(s1)	Returns the lowest index where s1 starts in this string, or -1 if s1 is not found in this string.
rfind(s1)	Returns the highest index where s1 starts in this string, or -1 if s1 is not found in this string.
count(subtring)	Returns the number of non-overlapping occurrences of this substring.

Converting Strings

Method	Description
capitalize()	Returns a copy of this string with only the first character capitalized.
lower()	Returns a copy of this string with all letters converted to lowercase.
upper()	Returns a copy of this string with all letters converted to uppercase.
title()	Returns a copy of this string with the first letter capitalized in each word.
swapcase()	Returns a copy of this string in which lowercase letters are converted to upper and uppercase to lowercase.
replace(old, new)	Returns a new string that replaces all the occurrence of the old string with a n string.
replace(old, new, n)	Returns a new string that replaces up to n number of the occurrence of the old with a new string.

Striping Whitespace Characters

Method	Description
lstrip()	Returns a string with the leading whitespace characters removed.
rstrip()	Returns a string with the trailing whitespace characters removed.
strip()	Returns a string with the starting and trailing whitespace characters removed.