17.1 String formatting (old)

Conversion specifiers

Program output commonly includes the values of variables as a part of the text. A ***string formatting expression*** allows a programmer to create a string with placeholders that are replaced by the values of variables. Such a placeholder is called a ***conversion specifier***, and different conversion specifiers are used to perform a conversion of the given variable value to a different type when creating the string.

Conversion specifiers convert the object into the desired type. Thus, if the programmer gives a float value of 5.5 to a '%d' conversion specifier, a truncated integer value of '5' is the result.

The syntax for using a conversion specifier also includes a % symbol between the string and the value or variable to be placed into the string. Ex:
`print('The couch is %d years old.' % couch_age)`

The '%%' sequence displays an actual percentage sign character, as in
`print('Annual percentage rate is %f%%' % apr)`.

PARTICIPATION ACTIVITY 17.1.1: Using string formatting expressions.

Animation captions:

1. Simple string replacement can be done using the %s conversion specifier.
2. The %d specifier is used for integer replacement in a formatted string.
3. The %f specifier is used for float replacement in a formatted string. If an integer is passed to a floating-point specifier, the value becomes a float.

Table 17.1.1: Common conversion specifiers.

Conversion specifier(s)	Notes	Example	Output

%d	Substitute as integer.	`print('%d' % 10)`	`10`
%f	Substitute as floating-point decimal	`print('%f' % 15.2)`	`15.200000`
%s	Substitute as string.	`print('%s' % 'ABC')`	`ABC`
%x, %X	Substitute as hexadecimal in lowercase (%x) or uppercase (%X).	`print('%x' % 31)`	`1f`
%e, %E	Substitute as floating-point exponential format in lowercase (%e) or uppercase (%E).	`print('%E' % 15.2)`	`1.520000E+01`

zyDE 17.1.1: Conversion specifiers automatically convert values.

The program below prints the average payday loan interest rate of 410.9
see Wikipedia: Payday loan). Try inputting the integer 411 instead, noting
converted by %f to 411.0.

Load default template...

```
410.9
```

Run

PARTICIPATION ACTIVITY 17.1.2: String formatting.

Complete the code using formatting specifiers to generate the described output.

1) Assume `price = 150`.

```
I saved $150!
```

```
print('I saved $
!' % price)
```

Check Show answer

2) Assume `percent = 40`.

```
Buy now! Save 40%!
```

```
print('Buy now! Save
!' % percent)
```

Check Show answer

Multiple conversion specifiers

Multiple conversion specifiers can appear within the string formatting expression. Expressions that contain more than one conversion specifier must specify the values within a tuple following the '%' character. The following print statement will print a sentence including two numeric values, indicated by the conversion specifiers %d and %f.

```
apr = float(input('Enter APR:\n'))

# Print using a float conversion specifier
print('Annual percentage rate as a float is %f

# Print using a decimal conversion specifier
print('Annual percentage rate as an int is %d

```

Figure 17.1.1: Multiple conversion specifiers.

```
years = 15
total = 500 * (years * 0.02)
print('Savings after %d years is: %f' % (years,
total))
```

```
Savings after 15 years is:
150.000000
```

PARTICIPATION ACTIVITY 17.1.3: Multiple conversion specifiers.

Complete the code using formatting specifiers to generate the described output. Use the indicated variables.

1) Assume `item = 'burrito'` and `price = 5`.

```
The burrito is $5
```

```
print('The ______ is
$%d' % (item, price))
```

Check Show answer

2) Assume `item = 'backpack'` and `weight = 5.2`.

```
The backpack is 5.200000
pounds.
```

```
print('The %s is
______ pounds.' %
(item, weight))
```

Check Show answer

3) Assume `city = 'Boston'` and `distance = 2100`.

```
We are 2100 miles from Boston.
```

```
print('We are %d miles from %s.' % (
                                   ))
```

Check Show answer

CHALLENGE ACTIVITY 17.1.1: Printing a string.

Write a *single* statement to print: user_word,user_number. Note that there is no space between the comma and user_number.

Sample output with inputs: 'Amy' 5

```
Amy,5
```

```
user_word = str(input())
user_number = int(input())

''' Your solution goes here '''

```

Run

View solution (Instructors only)

Download student submissions

CHALLENGE ACTIVITY 17.1.2: String formatting.

Start

Select the most appropriate conversion specifiers.

```
num_students = 24

print('The math class has %f students.' % num_students)
```

```
The math class has 24 students.
```

1 2 3

Check Next

View solution (Instructors only)

17.2 String formatting using dictionaries

Mapping keys

Sometimes a string contains many conversion specifiers. Such strings can be hard to read and understand. Furthermore, the programmer must be careful with the ordering of the tuple values, lest items are mistakenly swapped. A dictionary may be used in place of a tuple on the right side of the conversion operator to enhance clarity at the expense of brevity. If a dictionary is used, then all conversion specifiers must include a ***mapping key*** component. A mapping key is specified by indicating the key of the relevant value in the dictionary within parentheses.

PARTICIPATION ACTIVITY 17.2.1: Using a dictionary and conversion specifiers with mapping keys.

Animation captions:

1. A mapping key is specified by indicating the key of the relevant value in the dict within parentheses.

Figure 17.2.1: Comparing conversion operations using tuples and dicts.

```
import time
gmt = time.gmtime()  # Get current Greenwich Mean Time

print('Time is: {:02d}/{:02d}/{:04d}  {:02d}:{:02d} {:02d} sec' \
     .format(gmt.tm_mon, gmt.tm_mday, gmt.tm_year, gmt.tm_hour, gmt.tm_min,
gmt.tm_sec))
```

```
Time is: 06/07/2013  20:16 24 sec
...
Time is: 06/07/2013  20:16 28 sec
```

```
import time
gmt = time.gmtime()  # Get current Greenwich Mean Time

print('Time is: %(month)02d/%(day)02d/%(year)04d  %(hour)02d:%(min)02d %(sec)02d
sec' %  \
      {
        'year': gmt.tm_year, 'month': gmt.tm_mon, 'day': gmt.tm_mday,
        'hour': gmt.tm_hour, 'min': gmt.tm_min, 'sec': gmt.tm_sec
      }
)
```

```
Time is: 06/07/2013  20:16 24 sec
...
Time is: 06/07/2013  20:16 28 sec
```

PARTICIPATION ACTIVITY 17.2.2: Mapping keys.

Complete the print statement to produce the given output using mapping keys.

1) "I need 12 lilies, 6 roses, and 18 tulips."

```
print ('I need
%(lilies)d lilies,
%(roses)d roses, and
%(tulips)d tulips.' % {
                        })
```

Check **Show answer**

2) "My name is Jerome and I'm 15 years old."

```
print ('My name is
%(name)s and I am
%(age)d years old' % {
                        })
```

Check **Show answer**

17.3 zyBooks built-in programming window

zyDE 17.3.1: Programming window.

Load default template...

```
print('Enter your program here')

```

Pre-enter any input fo
run.

Run