

CSIT110

Fundamental Programming with Python

String Format

Goh X. Y.



In this lecture

Multi-line code statement with string

Escape sequence

String format


Numerical operations

Multi-line code statement

To end a statement in Python, you simply press Enter.
Therefore, this code will generate a syntax error:

```
subject_code = "CSCI111"  
subject_mark = 80  
subject_grade = "D"  
  
result = "Subject result: "  
    + subject_code  
    + " mark " + str(subject_mark)  
    + " grade " + subject_grade  
  
print(result)
```

Python thinks that this is
the end of the statement



Multi-line code statement

Use the backslash `\` to indicate that a statement is continued on the next line.

```
subject_code = "CSCI111"
subject_mark = 80
subject_grade = "D"

result = "Subject result: " \
    + subject_code \
    + " mark " + str(subject_mark) \
    + " grade " + subject_grade

print(result)
```

we can break a long line of code into multiple lines

Multi-line code statement

Line continuation is automatic when the split comes while a statement is inside parenthesis (, brackets [or braces {

Therefore, this code is fine:

```
subject_code = "CSCI111"
subject_mark = 80
subject_grade = "D"
print(
    "Subject result: "
    + subject_code
    + " mark " + str(subject_mark)
    + " grade " + subject_grade
)
```

Multi-line code statement

Sometimes, we should break a long line of code into multi-line to make it clearer

```
print(  
    "Subject result: "  
    + subject_code  
    + " mark " + str(subject_mark)  
    + " grade " + subject_grade  
)
```

```
print("Subject result: " + subject_code + " mark " +  
str(subject_mark) + " grade " + subject_grade)
```

Escape Sequence

```
print("Welcome to Unimovies!")  
print("Thursday July 30 at 7.15pm: Inside Out")
```

Program output:

```
Welcome to Unimovies!  
Thursday July 30 at 7.15pm: Inside Out
```

Escape Sequence

```
print("Welcome to Unimovies!")  
print("Thursday July 30 at 7.15pm: Inside Out")
```

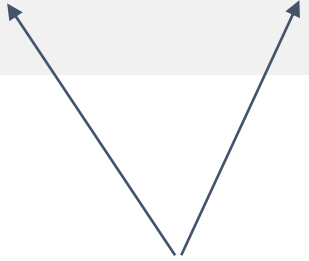
How do we write program for this output:

```
Welcome to Unimovies!  
Thursday July 30 at 7.15pm: "Inside Out"
```


Escape Sequence

How about this program?

```
print("Welcome to Unimovies!")  
print("Thursday July 30 at 7.15pm: "Inside Out")
```



what is wrong with this code?

We want to write a program for this output:

```
Welcome to Unimovies!  
Thursday July 30 at 7.15pm: "Inside Out"
```

Escape Sequence

The correct program

```
print("Welcome to Unimovies!")  
print("Thursday July 30 at 7.15pm: \"Inside Out\"")
```



using escape sequence

Program output:

```
Welcome to Unimovies!  
Thursday July 30 at 7.15pm: "Inside Out"
```

Alternatively

The correct program

```
print("Welcome to Unimovies!")  
print('Thursday July 30 at 7.15pm: "Inside Out"')
```

Use single quotes to embed double quotes
and vice versa (See 01PythonInputOutput slide 28.)

Program output:

```
Welcome to Unimovies!  
Thursday July 30 at 7.15pm: "Inside Out"
```

Escape Sequence

Escape Sequence	Meaning
\\	Backslash (\)
\'	Single quote (')
\"	Double quote (")
\n	New line
\t	Tab

Escape Sequence

```
print("Your details:\n")
print("\tName: \"John Smith\"")
print("\tSN:    \"2012345\"")
print("\nEnrolment record:\n")
print("\tMATH101")
print("\tCSCI201")
```

Program output:

```
Your details:

    Name: "John Smith"
    SN:   "2012345"

Enrolment record:

    MATH101
    CSCI201
```

Escape Sequence

```
print("Escape sequence:")
print("\\n : Insert a newline.")
print("\\t : Insert a tab.")
print("\\\"" : Insert a double quote character.")
print("\\'" : Insert a single quote character.")
print("\\\\" : Insert a backslash character.)
```

What is the output of this program?
Try it yourself!

String format – f-string

Formatting using f'...' or f'..."

{ } – things in the curly brackets are interpreted as code

Formatting numbers in string

{ :<8 } – sets minimum length of variable to 8 character

{ : .4% } – displays a float to 4 decimal places + converts number to a percentage

```
for_votes = 2843493
against_votes = 1223292
percentage = for_votes / (for_votes + against_votes)
print(f'{for_votes:<9} are for the notion ({percentage:.2%})')
```

```
2843493   are for the notion (69.92%)
```

String format with **alignment**

```
print("Alkali metals:")
print()
print(f'{"Element":<15}{"Symbol":<10}{"Atomic number":^25}{"Atomic weight":>15}')
```

```
print(f'{"Lithium":<15}{"Li":<10}{3:^25}{6.94:>15}')
```

```
print(f'{"Sodium":<15}{"Na":<10}{11:^25}{22.99:>15}')
```

```
print(f'{"Potassium":<15}{"K":<10}{19:^25}{39.098:>15}')
```

```
print(f'{"Rubidium":<15}{"Rb":<10}{37:^25}{85.468:>15}')
```

```
print(f'{"Caesium":<15}{"Cs":<10}{55:^25}{132.905:>15}')
```

```
print(f'{"Francium":<15}{"Fr":<10}{87:^25}{223:>15}')
```

```
print()
```

```
print("12345678901234567890123456789012345678901234567890123456789012345")
```

Program output

Alkali metals:

Element	Symbol	Atomic number	Atomic weight
Lithium	Li	3	6.94
Sodium	Na	11	22.99
Potassium	K	19	39.098
Rubidium	Rb	37	85.468
Caesium	Cs	55	132.905
Francium	Fr	87	223

12345678901234567890123456789012345678901234567890123456789012345


```

print("Alkali metals:")
print()
print(f'{"Element"   :<15}{"Symbol":<10}{"Atomic number":^25}{"Atomic weight":>15}')
```

```

print(f'{"Lithium"   :<15}{"Li":<10}{3 :^25}{6.94   :>15}')
```

```

print(f'{"Sodium"    :<15}{"Na":<10}{11:^25}{22.99   :>15}')
```

```

print(f'{"Potassium" :<15}{"K"  :<10}{19:^25}{39.098  :>15}')
```

```

print(f'{"Rubidium"   :<15}{"Rb":<10}{37:^25}{85.468  :>15}')
```

```

print(f'{"Caesium"    :<15}{"Cs":<10}{55:^25}{132.905 :>15}')
```

```

print(f'{"Francium"   :<15}{"Fr":<10}{87:^25}{223     :>15}')
```

```

print()
print("12345678901234567890123456789012345678901234567890123456789012345")
```

left alignment, using 15 spaces

Alkali metals:			
Element	Symbol	Atomic number	Atomic weight
Lithium	Li	3	6.94
Sodium	Na	11	22.99
Potassium	K	19	39.098
Rubidium	Rb	37	85.468
Caesium	Cs	55	132.905
Francium	Fr	87	223

12345678901234567890123456789012345678901234567890123456789012345

```

print("Alkali metals:")
print()
print(f'{"Element"   :<15}{"Symbol":<10}{"Atomic number":^25}{"Atomic weight":>15}')
```

```

print(f'{"Lithium"   :<15}{"Li":<10}{3 :^25}{6.94   :>15}')
```

```

print(f'{"Sodium"    :<15}{"Na":<10}{11:^25}{22.99   :>15}')
```

```

print(f'{"Potassium" :<15}{"K"  :<10}{19:^25}{39.098  :>15}')
```

```

print(f'{"Rubidium"   :<15}{"Rb":<10}{37:^25}{85.468   :>15}')
```

```

print(f'{"Caesium"    :<15}{"Cs":<10}{55:^25}{132.905  :>15}')
```

```

print(f'{"Francium"   :<15}{"Fr":<10}{87:^25}{223      :>15}')
```

```

print()
print("123456789012345678901234567890123456789012345678901234567890123456789012345")
```

left alignment, using 10 spaces

Alkali metals:			
Element	Symbol	Atomic number	Atomic weight
Lithium	Li	3	6.94
Sodium	Na	11	22.99
Potassium	K	19	39.098
Rubidium	Rb	37	85.468
Caesium	Cs	55	132.905
Francium	Fr	87	223

123456789012345678901234567890123456789012345678901234567890123456789012345

```

print("Alkali metals:")
print()
print(f'{"Element"      :<15}{"Symbol":<10}{"Atomic number":^25}{"Atomic weight":>15}')
```

```

print(f'{"Lithium"      :<15}{"Li":<10}{3 :^25}{6.94      :>15}')
```

```

print(f'{"Sodium"       :<15}{"Na":<10}{11:^25}{22.99     :>15}')
```

```

print(f'{"Potassium"    :<15}{"K"  :<10}{19:^25}{39.098    :>15}')
```

```

print(f'{"Rubidium"     :<15}{"Rb":<10}{37:^25}{85.468     :>15}')
```

```

print(f'{"Caesium"      :<15}{"Cs":<10}{55:^25}{132.905    :>15}')
```

```

print(f'{"Francium"     :<15}{"Fr":<10}{87:^25}{223        :>15}')
```

```

print()
print("12345678901234567890123456789012345678901234567890123456789012345")
```

centre alignment, using 25 spaces

Alkali metals:			
Element	Symbol	Atomic number	Atomic weight
Lithium	Li	3	6.94
Sodium	Na	11	22.99
Potassium	K	19	39.098
Rubidium	Rb	37	85.468
Caesium	Cs	55	132.905
Francium	Fr	87	223

123456789012345678901234567890123456789012345678901234567890123456789012345


```

print("Alkali metals:")
print()
print(f'{"Element"      :<15}{"Symbol":<10}{"Atomic number":^25}{"Atomic weight":>15}')
print(f'{"Lithium"      :<15}{"Li":<10}{3 :^25}{6.94      :>15.3f}')
print(f'{"Sodium"       :<15}{"Na":<10}{11:^25}{22.99     :>15.3f}')
print(f'{"Potassium"    :<15}{"K" :<10}{19:^25}{39.098    :>15.3f}')
print(f'{"Rubidium"     :<15}{"Rb":<10}{37:^25}{85.468    :>15.3f}')
print(f'{"Caesium"      :<15}{"Cs":<10}{55:^25}{132.905   :>15.3f}')
print(f'{"Francium"     :<15}{"Fr":<10}{87:^25}{223       :>15.3f}')
print()
print("12345678901234567890123456789012345678901234567890123456789012345")

```

have exactly **3 digits** after the decimal places

Alkali metals:			
Element	Symbol	Atomic number	Atomic weight
Lithium	Li	3	6.940
Sodium	Na	11	22.990
Potassium	K	19	39.098
Rubidium	Rb	37	85.468
Caesium	Cs	55	132.905
Francium	Fr	87	223.000

12345678901234567890123456789012345678901234567890123456789012345

```

print("Alkali metals:")
print()
print(f'{"Element"      :<15}{"Symbol":<10}{"Atomic number":^25}{"Atomic weight":>15}\'')
print(f'{"Lithium"      :<15}{"Li":<10}{3 :^25}{6.94      :>15.4f}\'')
print(f'{"Sodium"       :<15}{"Na":<10}{11:^25}{22.99     :>15.4f}\'')
print(f'{"Potassium"    :<15}{"K"  :<10}{19:^25}{39.098    :>15.4f}\'')
print(f'{"Rubidium"     :<15}{"Rb":<10}{37:^25}{85.468     :>15.4f}\'')
print(f'{"Caesium"      :<15}{"Cs":<10}{55:^25}{132.905    :>15.4f}\'')
print(f'{"Francium"     :<15}{"Fr":<10}{87:^25}{223        :>15.4f}\'')
print()
print("12345678901234567890123456789012345678901234567890123456789012345")

```

Alkali metals:

Element	Symbol	Atomic number	Atomic weight
Lithium	Li	3	6.9400
Sodium	Na	11	22.9900
Potassium	K	19	39.0980
Rubidium	Rb	37	85.4680
Caesium	Cs	55	132.9050
Francium	Fr	87	223.0000

12345678901234567890123456789012345678901234567890123456789012345

```

print("Alkali metals:")
print()
print(f'{"Element"      :<15}{ "Symbol":<10}{ "Atomic number":^25}{ "Atomic weight":>15}')
print(f'{"Lithium"      :<15}{ "Li":<10}{3  :^25}{6.94      :>15.0f}')
print(f'{"Sodium"       :<15}{ "Na":<10}{11:^25}{22.99     :>15.0f}')
print(f'{"Potassium"    :<15}{ "K"  :<10}{19:^25}{39.098    :>15.0f}')
print(f'{"Rubidium"     :<15}{ "Rb":<10}{37:^25}{85.468     :>15.0f}')
print(f'{"Caesium"      :<15}{ "Cs":<10}{55:^25}{132.905    :>15.0f}')
print(f'{"Francium"     :<15}{ "Fr":<10}{87:^25}{223        :>15.0f}')
print()
print("12345678901234567890123456789012345678901234567890123456789012345")

```

Alkali metals:

Element	Symbol	Atomic number	Atomic weight
Lithium	Li	3	7
Sodium	Na	11	23
Potassium	K	19	39
Rubidium	Rb	37	85
Caesium	Cs	55	133
Francium	Fr	87	223

12345678901234567890123456789012345678901234567890123456789012345

String format with **alignment**

```
print("Alkali metals:")
print()
print(f'{"Element":<15} {"Symbol":<10}{"Atomic number":^25}{"Atomic weight":>15}')
```

```
print(f'{"Lithium":<15} {"Li":<10} {"3":^25} {"6.94":>15}')
```

```
print(f'{"Sodium":<15} {"Na":<10} {"11":^25} {"22.99":>15}')
```

```
print(f'{"Potassium":<15} {"K":<10} {"19":^25} {"39.098":>15}')
```

```
print(f'{"Rubidium":<15} {"Rb":<10} {"37":^25} {"85.468":>15}')
```

```
print(f'{"Caesium":<15} {"Cs":<10} {"55":^25} {"132.905":>15}')
```

```
print(f'{"Francium":<15} {"Fr":<10} {"87":^25} {"223":>15}')
```

```
print()
```

```
print("123456789012345678901234567890123456789012345678901234567890123456789012345")
```



Program output

Alkali metals:

Element	Symbol	Atomic number	Atomic weight
Lithium	Li	3	6.94
Sodium	Na	11	22.99
Potassium	K	19	39.098
Rubidium	Rb	37	85.468
Caesium	Cs	55	132.905
Francium	Fr	87	223

12345678901234567890123456789012345678901234567890123456789012345

String format with **alignment**

Insert non-formatting string on the **LEFT** hand side of the colon

```
print("Alkali metals:")
print()
print(f'{"Element"      :<15}{"Symbol":<10}{"Atomic number":^25}{"Atomic weight":>15}')
```

```
print(f'{"Lithium"      :<15}{"Li":<10}{3 :^25}{6.94      :>15}')
```

```
print(f'{"Sodium"       :<15}{"Na":<10}{11:^25}{22.99     :>15}')
```

```
print(f'{"Potassium"    :<15}{"K"  :<10}{19:^25}{39.098    :>15}')
```

```
print(f'{"Rubidium"     :<15}{"Rb":<10}{37:^25}{85.468     :>15}')
```

```
print(f'{"Caesium"      :<15}{"Cs":<10}{55:^25}{132.905    :>15}')
```

```
print(f'{"Francium"     :<15}{"Fr":<10}{87:^25}{223        :>15}')
```

```
print()
```

```
print("12345678901234567890123456789012345678901234567890123456789012345")
```

Program output

Alkali metals:

Element	Symbol	Atomic number	Atomic weight
Lithium	Li	3	6.94
Sodium	Na	11	22.99
Potassium	K	19	39.098
Rubidium	Rb	37	85.468
Caesium	Cs	55	132.905
Francium	Fr	87	223

12345678901234567890123456789012345678901234567890123456789012345

Another example

```
print(f"{1} x {5} = {1*5}")
print(f"{2} x {5} = {2*5}")
print(f"{3} x {5} = {3*5}")
print(f"{4} x {5} = {4*5}")
print(f"{5} x {5} = {5*5}")
print(f"{6} x {5} = {6*5}")
print(f"{7} x {5} = {7*5}")
print(f"{8} x {5} = {8*5}")
print(f"{9} x {5} = {9*5}")
print(f"{10} x {5} = {10*5}")
```

1	x	5	=	5
2	x	5	=	10
3	x	5	=	15
4	x	5	=	20
5	x	5	=	25
6	x	5	=	30
7	x	5	=	35
8	x	5	=	40
9	x	5	=	45
10	x	5	=	50

Another example

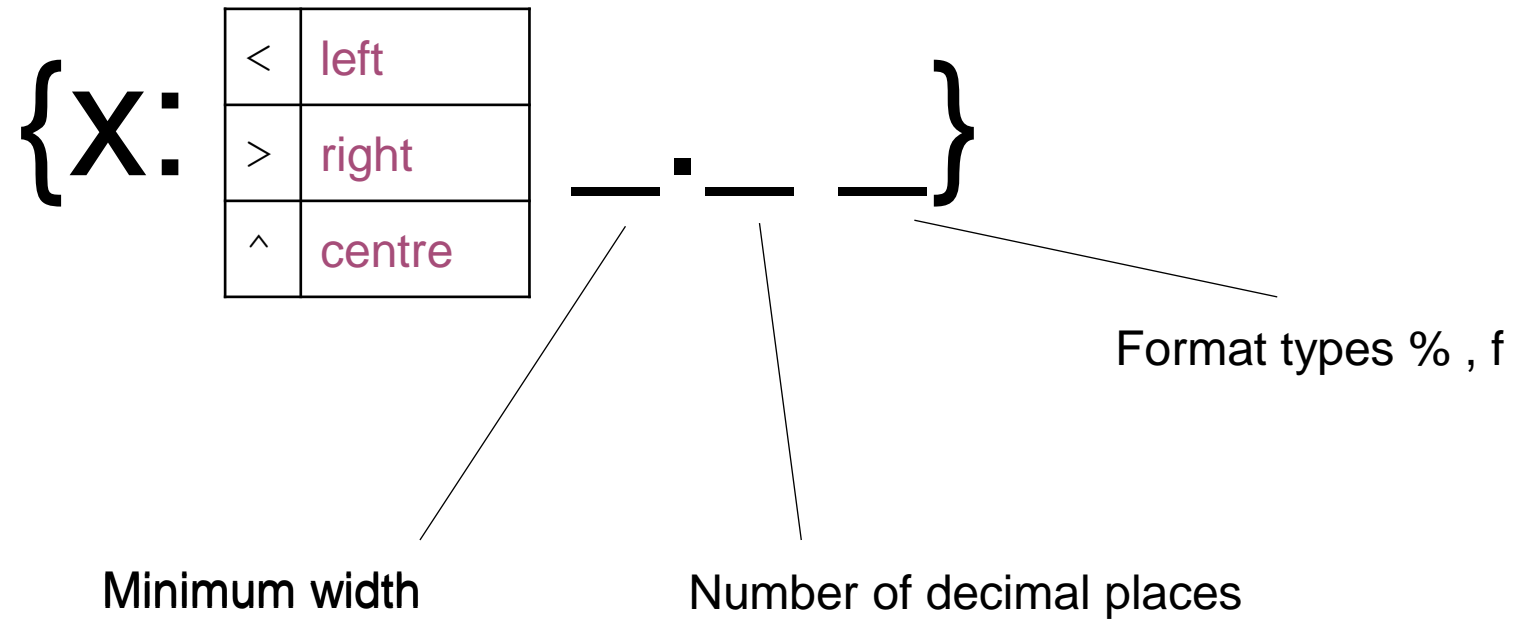
```
print(f"{1:>2} x {5:>1} = {1*5:>2}")
print(f"{2:>2} x {5:>1} = {2*5:>2}")
print(f"{3:>2} x {5:>1} = {3*5:>2}")
print(f"{4:>2} x {5:>1} = {4*5:>2}")
print(f"{5:>2} x {5:>1} = {5*5:>2}")
print(f"{6:>2} x {5:>1} = {6*5:>2}")
print(f"{7:>2} x {5:>1} = {7*5:>2}")
print(f"{8:>2} x {5:>1} = {8*5:>2}")
print(f"{9:>2} x {5:>1} = {9*5:>2}")
print(f"{10:>2} x {5:>1} = {10*5:>2}")
```

we want a better output

1	x	5	=	5
2	x	5	=	10
3	x	5	=	15
4	x	5	=	20
5	x	5	=	25
6	x	5	=	30
7	x	5	=	35
8	x	5	=	40
9	x	5	=	45
10	x	5	=	50

Part 2

Recap



Using quotation marks in f-string

```
1 x = "some text"
2 print(f'Here is {x} ')
3 print(f'Here is {x + 'and more text'}')
4 print(f'Here is {x + "and more text"}')
5 print(f'Here is {x + "with a quote \"text\""}')
6 y = '\"Never give up \">'
7 print(f'Here is {x + " and a quote " + y}')
8 print(f'Here is {x} and a quote {y}')
```

Here is some text



Here is some text and more text



Here is some text and a quote "Never give up"

Using quotation marks in f-string

1

```
x = "some text"
```

2

```
print(f'Here is {x}')
```

Here is some text

Using quotation marks in f-string

```
1 x = "some text"  
2 print(f'Here is {x} ')  
3 print(f'Here is {x + 'and more text'}')
```

Here is some text



Using quotation marks in f-string

```
1 x = "some text"
2 print(f'Here is {x} ')
3 print(f'Here is {x + 'and more text'}')
4 print(f'Here is {x + "and more text"}')
```

Here is some text



Here is some text and more text

Using quotation marks in f-string

```
1 x = "some text"
2 print(f'Here is {x} ')
3 print(f'Here is {x + 'and more text'}')
4 print(f'Here is {x + "and more text"}')
5 print(f'Here is {x + "with a quote \"text\""}')
```

Here is some text

✗

Here is some text and more text

✗

Using quotation marks in f-string

```
1 x = "some text"
2 print(f'Here is {x} ')
3 print(f'Here is {x + 'and more text'}')
4 print(f'Here is {x + "and more text"}')
5 print(f'Here is {x + "with a quote \"text\""}')
6 y = '\nNever give up \n'
```

Here is some text



Here is some text and more text



Using quotation marks in f-string

```
1 x = "some text"
2 print(f'Here is {x} ')
3 print(f'Here is {x + 'and more text'}')
4 print(f'Here is {x + "and more text"}')
5 print(f'Here is {x + "with a quote \"text\""}')
6 y = '\"Never give up \">'
7 print(f'Here is {x + " and a quote " + y}')
```

Here is some text



Here is some text and more text



Here is some text and a quote "Never

Using quotation marks in f-string

```
1 x = "some text"
2 print(f'Here is {x} ')
3 print(f'Here is {x + 'and more text'}')
4 print(f'Here is {x + "and more text"}')
5 print(f'Here is {x + "with a quote \"text\""}')
6 y = '\"Never give up \"'
7 print(f'Here is {x + " and a quote " + y}')
8 print(f'Here is {x} and a quote {y}')
```

Here is some text



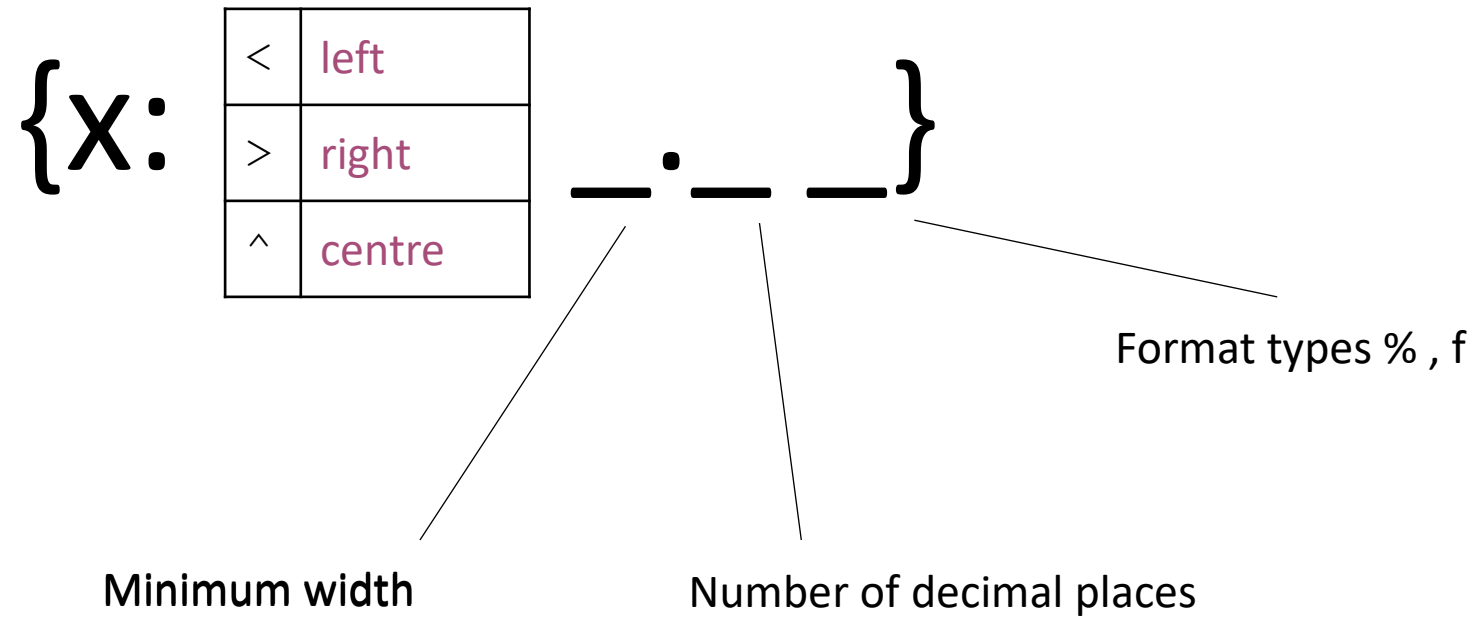
Here is some text and more text



Here is some text and a quote "Never give up"

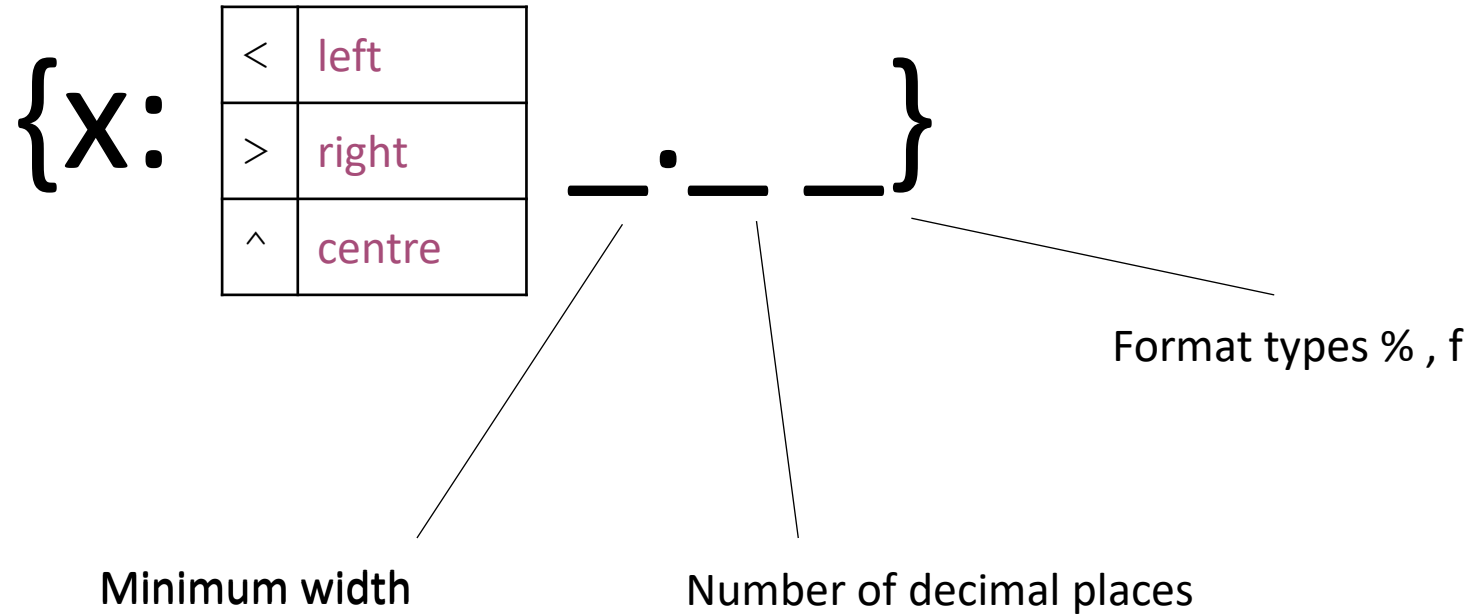
Part 3

Previously



Set width with a variable

```
width = 3  
dp = 4
```



Set width with a variable

```
width = 3  
dp = 4
```

{x:

<	left
>	right
^	centre

{width}.{dp}f}

Minimum width

Number of decimal places

Format types %, f

Any questions?