PYTHON

LESSON 1

CODE FIRST GIRLS

NANODEGREE → FOUNDATION MODULE

AGENDA

01 Introduction to Python

02 Numbers and Operators

03 String Data Type

04 Variables

05 ASCII

06 Slicing

07 In-built functions

PYTHON

KEY FEATURES

- **Easy to code:** Python is very easy to learn the language as compared to other languages
- **Free and Open Source:** It means that Python source code is freely available to the public for download
- **Object-Oriented Language:** One of the key features of python is Object-Oriented programming
- **High-Level Language:** When we write programs in python, we do not need to remember the system architecture, nor do we need to manage the memory.
- **Extensible feature:** We can write us some Python code into C or C++ language
- **Python is Portable language:** Python code for windows can be run on other platforms such as Linux, Unix, and Mac
- **Interpreted Language:** Python code is executed line by line at a time, there is no need to compile python code.
- **Dynamically Typed Language:** That means the type for a variable is decided at run time not in advance, so we don't need to specify the type of variable.

FUNCTION

DEFINITION

- **Function:** A reusable piece of code that completes a specific task

- You can recognise a function as they are a word followed by round brackets `()` e.g. `print()`

- The print() function is used to output a message to the programmer

- You can change the data given to the function to change the output

```
# EXAMPLE

print('I hope it is sunny this weekend')
```

NUMBERS AND OPERATORS

DEFINITION

- **Integer:** a Python **data type** for **whole numbers**. For example `5`, `-99` and `1048` are all integers.

- **Float:** a Python **data type** for **decimal numbers**. For example `5.6`, `9.0` and `-67.1001` are all floats.

EXAMPLE

Operator types

- `+`: add
- `-`: subtract
- `*`: multiply
- `/`: division
- `**`: exponent
- `%`: modulo (remainder)

STRING DATA TYPE

DEFINITION

- **String:** a Python data type for **text** and **characters**.
- For example Strings must be written between a pair of single or double speech marks
- `'...'` or `"..."`
- Forgetting the speechmark will cause exception
- The * and + operators work on strings as well as integers.

EXAMPLE

- `'Hello'`
- `"Abcdef1234"#`
- `'cats'`

STRING DATA TYPE

METHODS

- **method:** A repeatable piece of code that completes a task for specific data-type

- Methods are like functions, but they are tied to specific data-types e.g. `.upper()` can only used with a string and not an integer or a float

EXAMPLE

- `.upper()`

- `.lower()`

- `.title()`

VARIABLES

DEFINITION

- **Variable:** a reusable **label** for a data value in Python

- Creating (assigning) a variable has three parts:

    1. The variable's name
    2. An equals sign =
    3. The data value it references

EXAMPLE

- username = **'Jenny_1995'**
- age = **23**

VARIABLES

DEFINITION

- Values and variables are interchangeable

- A variable can be put anywhere that a data value can be used

EXAMPLE

- print('spaghetti')

- food = 'spaghetti'
- print(food)

STRING FORMATTING

FORMAT METHOD

- Compare these examples:

```
oranges = 12

cost_per_orange = 0.5

total_cost = oranges *
cost_per_orange

output = str(oranges) + " oranges
costs £" + str(total_cost)

print(output)
```

EXAMPLE

- Python strings have a method (.format()) that substitutes place-holders {} for values

```
oranges = 12

cost_per_orange = 0.5

total_cost = oranges *
cost_per_orange

output = "{} oranges costs
£{}".format(oranges, total_cost)

print(output)
```

STRING FORMATTING

JOIN METHOD

- We need to have few strings to join up together
- And for now, let's say we are going to put our strings inside a container with [ ] brackets
- We need to specify the string symbol that we are going to use to perform the join

EXAMPLE

CODE:

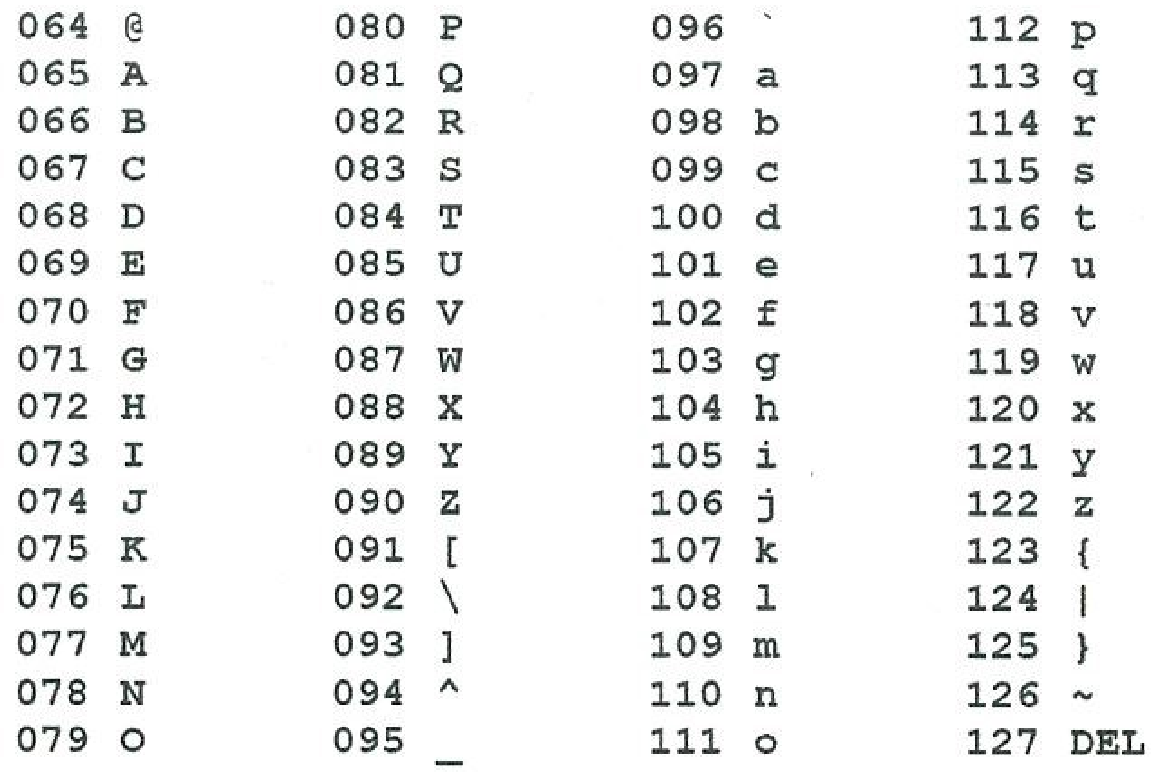
```
my_words = ['apple', 'banana', 'kiwi']

', '.join(my_words)
```

RESULT:

```
'apple, banana, kiwi'
```

ASCII

AMERICAN STANDARD CODE FOR INFORMATION interchange



<table>
<tr><td>064</td><td>@</td><td>080</td><td>P</td><td>096</td><td>`</td><td>112</td><td>p</td></tr>
<tr><td>065</td><td>A</td><td>081</td><td>Q</td><td>097</td><td>a</td><td>113</td><td>q</td></tr>
<tr><td>066</td><td>B</td><td>082</td><td>R</td><td>098</td><td>b</td><td>114</td><td>r</td></tr>
<tr><td>067</td><td>C</td><td>083</td><td>S</td><td>099</td><td>c</td><td>115</td><td>s</td></tr>
<tr><td>068</td><td>D</td><td>084</td><td>T</td><td>100</td><td>d</td><td>116</td><td>t</td></tr>
<tr><td>069</td><td>E</td><td>085</td><td>U</td><td>101</td><td>e</td><td>117</td><td>u</td></tr>
<tr><td>070</td><td>F</td><td>086</td><td>V</td><td>102</td><td>f</td><td>118</td><td>v</td></tr>
<tr><td>071</td><td>G</td><td>087</td><td>W</td><td>103</td><td>g</td><td>119</td><td>w</td></tr>
<tr><td>072</td><td>H</td><td>088</td><td>X</td><td>104</td><td>h</td><td>120</td><td>x</td></tr>
<tr><td>073</td><td>I</td><td>089</td><td>Y</td><td>105</td><td>i</td><td>121</td><td>y</td></tr>
<tr><td>074</td><td>J</td><td>090</td><td>Z</td><td>106</td><td>j</td><td>122</td><td>z</td></tr>
<tr><td>075</td><td>K</td><td>091</td><td>[</td><td>107</td><td>k</td><td>123</td><td>{</td></tr>
<tr><td>076</td><td>L</td><td>092</td><td>\</td><td>108</td><td>l</td><td>124</td><td>|</td></tr>
<tr><td>077</td><td>M</td><td>093</td><td>]</td><td>109</td><td>m</td><td>125</td><td>}</td></tr>
<tr><td>078</td><td>N</td><td>094</td><td>^</td><td>110</td><td>n</td><td>126</td><td>~</td></tr>
<tr><td>079</td><td>O</td><td>095</td><td>_</td><td>111</td><td>o</td><td>127</td><td>DEL</td></tr>
</table>

ASCII

AMERICAN STANDARD CODE FOR INFORMATION interchange

EXAMPLE

ord()

- Python ord() function takes string argument of a single Unicode character and returns its integer Unicode code point value. In other words, the function accepts any ONE character or symbol and tells us what which number in ASCII it corresponds to.

chr()

- Python chr() function takes integer argument and returns the string representing a character at that code point. It is a reverse to ord() function

CODE:

```
ord('P')

# it is case sensitive! Uncomment and try below

#ord('p')
```

CODE:

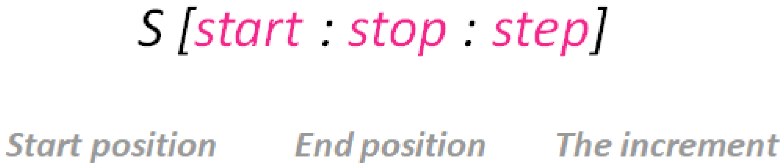
```
chr(86)
```

STRING SLICING

SLICING CREATES SUBSTRING

- Python string uses slicing to create substring.
- Slicing creates a new substring from the source string and the original string remains unchanged.
- Think of it as a way to extract specific characters from or manipulate given text.

EXAMPLE



S [start : stop : step]

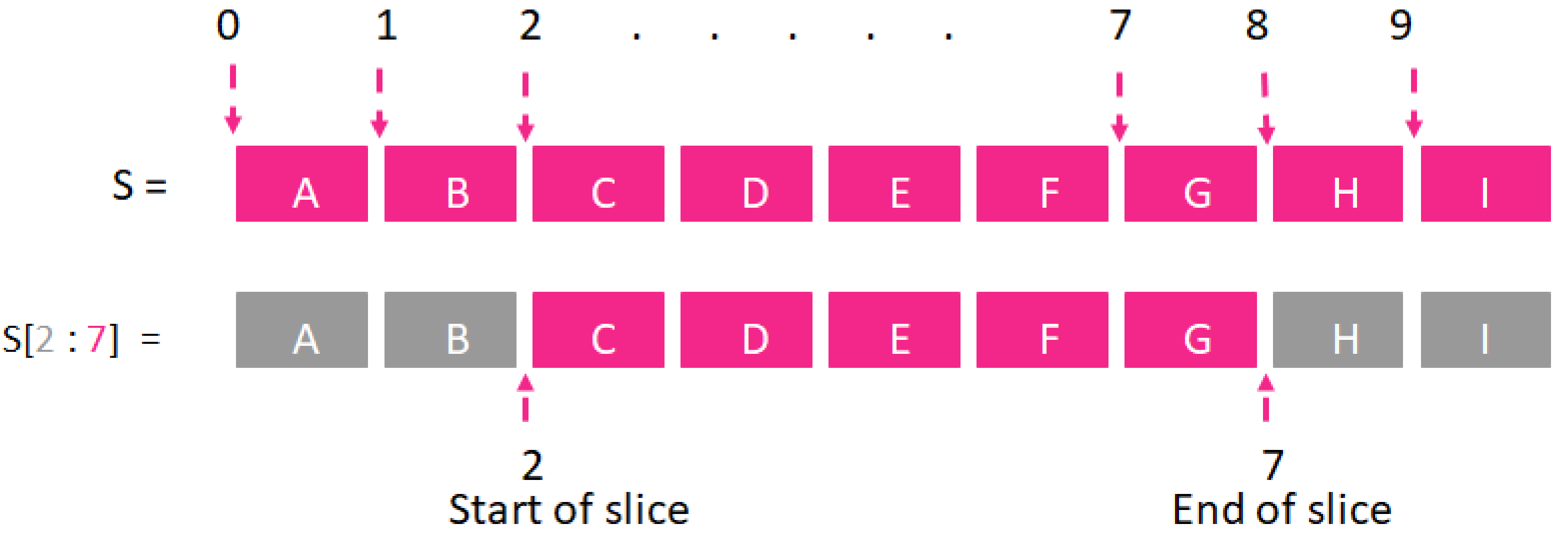
Start position   End position   The increment

NOTES:

1. **step** is an optional parameter and is defaulted to 1
2. you can omit 'step' from the syntax [start : stop]

STRING SLICING

SLICING CREATES SUBSTRING

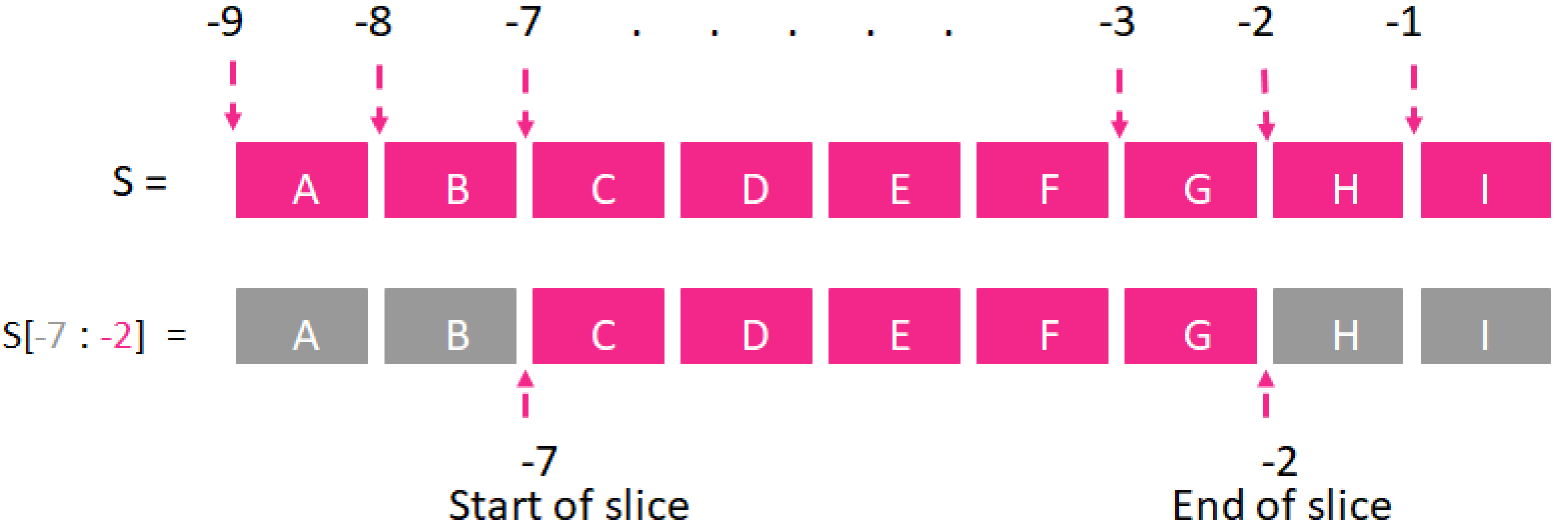


NOTES:

1. **index** starts from **0**
2. the **stop** number is NOT included

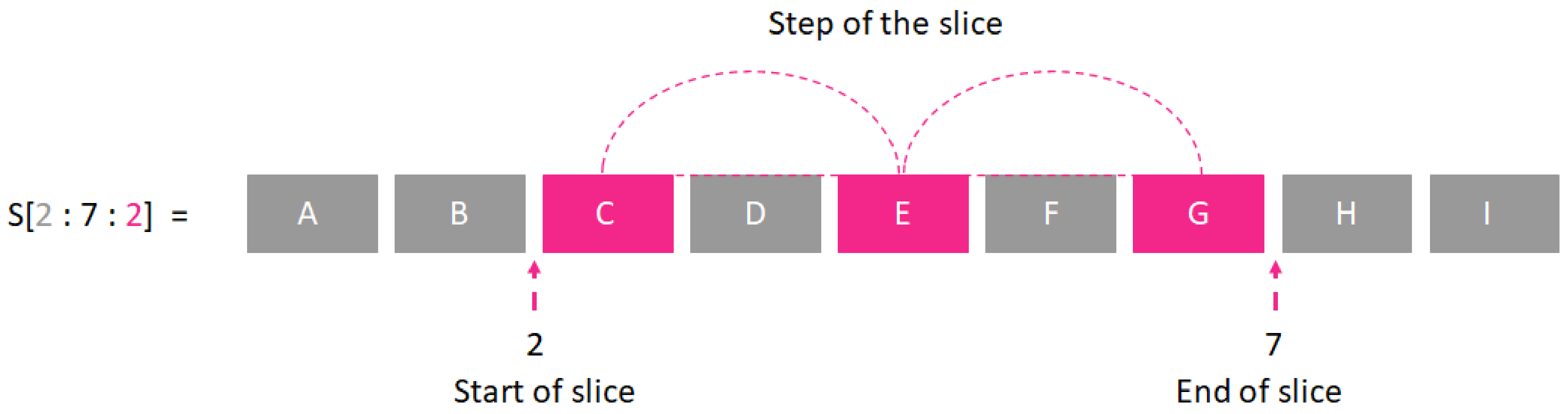
STRING SLICING

NEGATIVE INDEX



STRING SLICING

POSITIVE & NEGATIVE INDEX



Step of the slice

S[2 : 7 : 2] =

A	B	C	D	E	F	G	H	I

2
Start of slice

7
End of slice

IN BUILT FUNCTION

PYTHON

- Functions that are provided 'for free' with Python
- They are useful and we can just use them whenever we want
- There are lots of functions available and we already know some of them

NB: functions and methods (like `.upper()` that we used for strings) are two different things. We will learn about the difference later in this course.

# EXAMPLE

- `print()` `help()` `type()`
- `str()` `float()` `int()`
- `ord()` `chr()`
- `len()`

CODE
FIRST
GIRLS

THANK YOU!