

# CSIT881

## Programming and Data Structures

**Input & Output**



UNIVERSITY  
OF WOLLONGONG  
AUSTRALIA

# Objectives

- Variables & Data types
- Convert between data types
- Input and Output
  - `print` function
  - `input` function

# Our first Python programs

```
# My first Python program
print("PPP      Y      Y      TTTTT      H      H      OO      N      N")
print("P      P      Y Y      T      H      H      O      O      NN      N")
print("PPP          Y          T      HHHH      O      O      N      N      N")
print("P          Y          T      H      H      O      O      N      NN")
print("P          Y          T      H      H      OO      N      N")

# print blank lines
print()
print()

# print greetings
print("Welcome to Python!")
```

What do you think this program will do?

Write this python code and run it.  
See what the code produces.



# Our first Python programs

```
# print hello and greeting
print("Hello World!")
print('Welcome to Python!')
```

```
# print hello and greeting and silly stuff :-)
print("Hello World!", end="frog")
print("Welcome to Python!", end="cat")
print("How are you?")
```

## What is the purpose of

```
print("...")
print('...')
print("...", end="...")
print()
```

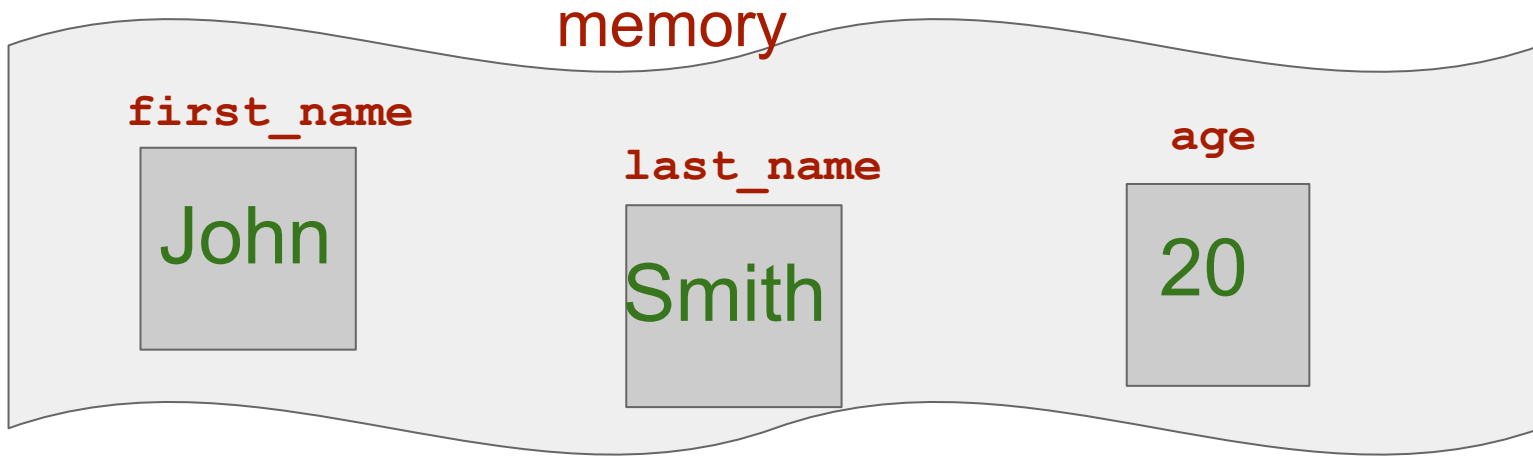


## What is wrong with this code?

```
print(Hello World!)
```

# Variables & Data types

**Variables** are reserved memory locations to store values



**ALWAYS** use variables with **meaningful names** and **correct data types**

```
first_name = "John"  
last_name = "Smith"  
age = 20
```

**NEVER** use variable like a, b, c, x, y, z, or blah...



# Variables & Data types

Variables store values in certain data types. Common data types:

- **str**: a string represents a sequence of characters.

We use double quotes or single quotes to create a string.

```
first_name = "John"  
state = 'New South Wales'
```

- **int**: an integer, a whole number

```
age = 20
```

6

- **float**: a decimal number

```
interest_rate = 5.2
```

- **bool**: a boolean value is either True or False.

```
scan_completed = True  
virus_found = False
```

# Variables & Data types

Each variable has a data type.

Checking data type: **type**(...variable\_name...)

**String:** using either double quote or single quote

```
first_name = "John"  
last_name = 'Smith'
```

```
print(type(first_name))  
print(type(last_name))
```

```
<class 'str'>
```

# Variables & Data types

**Integer:** whole numbers

```
age = 20
```

```
temperature = -5
```

```
credit_point = 6
```

```
print(type(age))
```

```
print(type(temperature))
```

```
print(type(credit_point))
```

```
<class 'int'>
```



# Variables & Data types

**Float:** decimal numbers

```
price = 30.5
interest_rate = 3.18

print(type(price))
print(type(interest_rate))
```

```
<class 'float'>
```

**Some important math constants**

```
import math

pi = math.pi
e = math.e
tau = math.tau
```

```
print(pi)
print(e)
print(tau)
```

```
3.141592653589793
2.718281828459045
6.283185307179586
```

# Variables & Data types

**Boolean:** True or False

```
virus_scan_completed = True  
virus_found = False
```

```
print(type(virus_scan_completed))  
print(type(virus_found))
```

```
<class 'bool'>
```

**Example:**

```
temperature = -5
```

```
temperature_negative = (temperature < 0)  
print(temperature_negative)
```

**True**

```
temperature_positive = (temperature > 0)  
print(temperature_positive)
```

**False**

# Variables & Data types

**Date data type:** including year, month, day, (not the time)

```
import datetime
```

```
today_date = datetime.date.today()
```

```
us_election_2020 = datetime.date(2020, 11, 3)
```

```
print(type(today_date))
```

```
print(type(us_election_2020))
```

```
<class 'datetime.date'>
```

# Variables & Data types

**Date-time data type:** including year, month, day, hour, minute, second, ...

```
import datetime
```

```
current_date_time = datetime.datetime.now()
```

```
christmas_2020 = datetime.datetime(2020, 12, 25)
```

```
random_date_time = datetime.datetime(2000, 12, 20, 14, 20, 39, 555)
```

```
print(type(current_date_time))
```

```
print(type(christmas_2020))
```

```
print(type(random_date_time))
```

```
<class 'datetime.datetime'>
```

# Important programming rules



## Variable contains data information only

Bad example:

```
subject = "MATH111: Abstract Algebra"
```

*The colon (:) is not part of the information and should not be stored in variable.  
What if we want to display like this:*

MATH111 - Abstract Algebra

*or this:*

Abstract Algebra (MATH111)

Good example:

```
subject_code = "MATH111"  
subject_title = "Abstract Algebra"
```

# Important programming rules



## Variable must be in correct data type

Bad example:

```
unit_price = "$10.50"
```

*Unit price should be a number, not a string.*

Good example:

```
unit_price = 10.50  
quantity = 12  
cost = unit_price * quantity
```

# Important programming rules



## Variable must be in correct data type

Bad example:

```
mobile_number = 1231231234  
student_number = 1234567
```

*Mobile number should be a string, not a number.*

*Student number should be a string, not a number.*

Good example:

```
mobile_number = "0980980987"  
student_number = "0043210"
```

# String addition (concatenation)

```
# name details
first_name = "John"
last_name = "Smith"

# use string addition to formulate the full name
full_name = first_name + " " + last_name

# display the full name
print("My name is " + full_name + ".")
```

My name is John Smith.



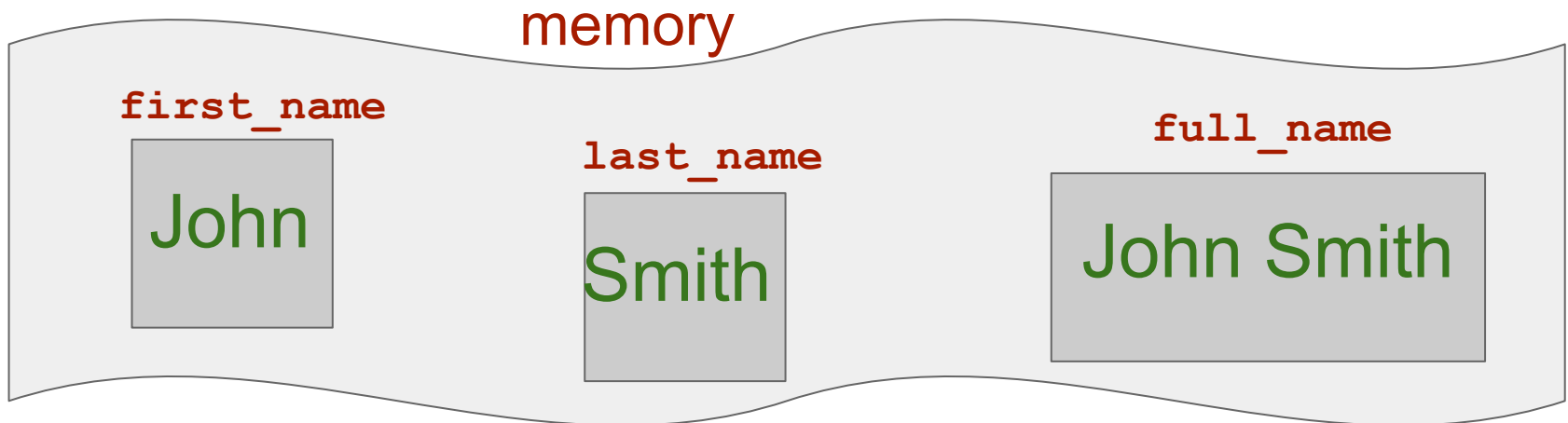
# String addition (concatenation)

```
# name details
first_name = "John"
last_name = "Smith"

# use string addition to formulate the full name
full_name = first_name + " " + last_name

# display the full name
print("My name is " + full_name + ".")
```

this is called  
**string addition**  
**(concatenation)**



# String multiplication (with number)!

```
# display some silly strings

silly1 = "frog" * 7

silly2 = 5 * "I am Sam "

print(silly1)

print(silly2)
```

frogfrogfrogfrogfrogfrogfrog

I am Sam I am Sam I am Sam I am Sam I am Sam

# Get input from the user

```
# ask the user to enter first name and last name
first_name = input("Enter your first name: ")
last_name = input("Enter your last name: ")

# use string addition to formulate the full name
full_name = first_name + " " + last_name

# display the full name
print("My name is " + full_name + ".")
```

```
Enter your first name: Mary
Enter your last name: Wilson
My name is Mary Wilson.
```

# Get input from the user

```
# ask the user to enter some information  
variable_here = input("Put the prompt here: ")
```

When we want to ask the user some information, use the `input` function.

In the `input` function, we can specify the **prompt**.

The information that the user has entered will be stored in the **variable** as a **string**.



# Get input from the user

```
# Ask the user to enter 3 subjects
print("You must choose 3 subjects.")
print()

subject1 = input("Enter the 1st subject: ")
subject2 = input("Enter the 2nd subject: ")
subject3 = input("Enter the 3rd subject: ")

# Display subjects
print()
print("You have chosen: " + subject1 + ", " + subject2 + ", " + subject3 + ".")
```

You must choose 3 subjects.

Enter the 1st subject: **ISIT111**

Enter the 2nd subject: **MATH101**

Enter the 3rd subject: **ACCY113**

You have chosen: ISIT111, MATH101, ACCY113.

# Get input from the user

```
# Ask the user to enter 3 subjects
print("You must choose 3 subjects.")
print()

subject1 = input("Enter the 1st subject: ")
subject2 = input("Enter the 2nd subject: ")
subject3 = input("Enter the 3rd subject: ")

# Display subjects
print()
print("You have chosen: "
      + subject1 + ", "
      + subject2 + ", "
      + subject3 + ".")
)
```

Rewrite the code to make it clearer.

*When we have a lot of string additions,  
write it this way make the code clearer!*



# Convert number into string

```
# A program to display a favorite number

# favorite number
fav_number = 7

# display favorite number
print("My favorite number is " + fav_number)
```

Write this python code and run it.

You will see that the code cannot run because there is an error.

What is wrong with this code?



# Convert number into string

```
# A program to display a favorite number

# favorite number
fav_number = 7

# display favorite number
print("My favorite number is " + fav_number)
```

*this is a string*

*this is a number*

Python cannot add a string to a number

(some other programming languages can)



# Convert number into string

```
# A program to display the favorite number  
  
# favorite number  
fav_number = 7  
  
# display favorite number  
print("My favorite number is " + str(fav_number))
```

convert a number to a string



fav\_number → 7

str(fav\_number) → "7"

now we can do string addition

"My favorite number is " + "7"

My favorite number is 7

# Convert string into number

```
# Ask the user to enter 2 integers and display the sum
number1 = input("Enter the 1st integer: ")
number2 = input("Enter the 2nd integer: ")

# calculate the sum
number_sum = number1 + number2

# display the sum
print("The sum is " + number_sum)
```

```
Enter the 1st integer: 100
Enter the 2nd integer: 50
The sum is 10050
```

why the output is like this



# Convert string into number

```
# Ask the user to enter 2 integers and display the sum
number1 = input("Enter the 1st integer: ")
number2 = input("Enter the 2nd integer: ")

# calculate the sum
number_sum = number1 + number2

# display the sum
print("The sum is " + number_sum)
```

```
Enter the 1st integer: 100
Enter the 2nd integer: 50
The sum is 10050
```

```
number1 is a string "100"
number2 is a string "50"
string addition means
number_sum is a string "10050"
```

When we ask the user to enter an input, then this input is a **string**.



# Convert string into number

```
# Ask the user to enter 2 integers and display the sum
user_input1 = input("Enter the 1st integer: ")
number1 = int(user_input1)

user_input2 = input("Enter the 2nd integer: ")
number2 = int(user_input2)

# calculate the sum
number_sum = number1 + number2

# display the sum
print("The sum is " + str(number_sum))
```

```
Enter the 1st integer: 100
Enter the 2nd integer: 50
The sum is 150
```

What did we change?



# Convert string into number

```
# Ask the user to enter 2 integers and display the sum
user_input1 = input("Enter the 1st integer: ")
number1 = int(user_input1)

user_input2 = input("Enter the 2nd integer: ")
number2 = int(user_input2)

# calculate the sum
number_sum = number1 + number2

# display the sum
print("The sum is " + str(number_sum))
```

user\_input1 is a **string** "100"  
number1 is an **integer number**

user\_input2 is a **string** "50"  
number2 is an **integer number**

```
Enter the 1st integer: 100
Enter the 2nd integer: 50
The sum is 150
```

number addition means number\_sum is a **number** 150

# Convert string into number

```
# Ask the user to enter 2 integers and display the sum
user_input = input("Enter the 1st integer: ")
number1 = int(user_input)

user_input = input("Enter the 2nd integer: ")
number2 = int(user_input)

# calculate the sum
number_sum = number1 + number2

# display the sum
print("The sum of "
      + str(number1)
      + " and "
      + str(number2)
      + " is "
      + str(number_sum)
      )
```

```
Enter the 1st integer: 100
Enter the 2nd integer: 50
The sum of 100 and 50 is 150
```

We can use just one variable `user_input` to save memory

# Convert string into decimal number

```
# Ask the user to enter 2 decimal numbers and display the sum
user_input = input("Enter the 1st number: ")
number1 = float(user_input)

user_input = input("Enter the 2nd number: ")
number2 = float(user_input)

# calculate the sum
number_sum = number1 + number2

# display the sum
print("The sum of "
      + str(number1)
      + " and "
      + str(number2)
      + " is "
      + str(number_sum)
      )
```

```
Enter the 1st number: 2.5
Enter the 2nd number: 3.1
The sum of 2.5 and 3.1 is 5.6
```

We use `number1 = float(user_input)` to convert the string `user_input` into a decimal number `number1`

# Get input from the user

```
# ask the user to enter some information  
variable_here = input("Put the prompt here: ")
```

If the information we need from the user is a **string**, then we just need one line of code:

```
# ask the user to enter city name  
city = input("Please enter the city name: ")
```

```
# ask the user to enter the name of a song  
song_name = input("Enter a song title: ")
```

```
# ask the user to enter the job title  
job = input("Please enter your job title: ")
```



# Get input from the user

```
# ask the user to enter some information  
variable_here = input("Put the prompt here: ")
```

If the information we need from the user is not a string, but of **other data types**, then we need

- one line of code: to get user input string
- another line of code: to convert into correct data type

```
# ask the user to enter the year of birth  
user_input = input("Please enter your year of birth: ")  
year = int(user_input)
```

```
# ask the user to enter the interest rate  
user_input = input("Enter the interest rate: ")  
rate = float(user_input)
```

# Convert between data types



**Convert to a string:** `str(...variable_name...)`

`fav_number`                       $\longrightarrow$       7

`str(fav_number)`                       $\longrightarrow$       "7"

`str()` can be used to convert other data types into string, such as boolean, list, dictionary ect.

**Convert to an integer:** `int(...variable_name...)`

```
user_input = input("Enter an integer: ")
number = int(user_input)
```

`user_input`                       $\longrightarrow$       "50"

`int(user_input)`                       $\longrightarrow$       50

# Convert between data types

**Convert to a decimal number:** `float(...variable_name...)`

`input1`                       $\longrightarrow$                       `"2.3"`

`float(input1)`                       $\longrightarrow$                       `2.3`

We can also convert integer to float, float to integer, etc...

# Convert between data types

## Convert between string and date

```
import datetime
```

```
# ask the user enter dob in DD/MM/YYYY format
```

```
user_input = input("Enter your dob (DD/MM/YYYY): ")
```

```
# convert string type to date type
```

```
date_format = '%d/%m/%Y'
```

```
dob = datetime.datetime.strptime(user_input, date_format).date()
```

```
# convert date to string
```

```
print("Your dob is " + dob.strftime("%d/%b/%Y"))
```

```
print("Your dob is " + dob.strftime("%d-%m-%Y"))
```

```
Enter your dob (DD/MM/YYYY): 26/03/2000
```

```
Your dob is 26/Mar/2000
```

```
Your dob is 26-03-2000
```

# Convert between data types

We can transform value from one data type to another data type. This is called “type casting”.

- Convert to string: `str(...)`
- Convert to integer: `int(...)`
- Convert to float: `float(...)`
- Convert to boolean: `bool(...)`

# Naming convention

```
first_name = "John"
last_name = "Smith"
full_name = first_name + " " + last_name

fav_number = 7

subject1 = "ISIT111"
subject2 = "MATH101"
subject3 = "ACCY113"

SECOND_PER_MINUTE = 60

minute = 5
second = minute * SECOND_PER_MINUTE
```

**ALWAYS** use variables with **meaningful names**

lower\_case\_with\_underscores for normal variables

UPPER\_CASE\_WITH\_UNDERSCORES for constant



# Comments

```
# print blank lines
```



comment

```
print()
```

```
print()
```

```
# print greetings
```



comment

```
print("Welcome to Python - Class of 2020!")
```

We can put comments anywhere in the program:

- to **make the program clearer** for people to read and maintain
- to **help people understand** our program better, especially, if our program has a special logic that needs explanation
- comments are not code, so they will NOT be executed

# Important programming rules



**ALWAYS** write comments first, then code.

**NEVER** write code first, then insert comments.

**ALWAYS** use variables with **meaningful names**

**NEVER** use variable like a, b, c, x, y, z, or blah...



# Keywords

The following list shows the Python keywords. These are reserved words and we CANNOT use them as constant or variable or any other identifier names.

and	elif	if	print
as	else	import	raise
assert	except	in	return
break	exec	is	try
class	finally	lambda	while
continue	for	not	with
def	from	or	yield
del	global	pass	