Week 10 Tutorial

COMP10001 – Foundations of Computing

Semester 2, 2025

Clement Chau

- Files I/O
- CSV Files
- Exceptions
- Computational Counting: Binary, Decimal, Octal, Hexadecimal

Agenda

1. Week 10 Discussion – **Tutorial sheet** (~ 65 mins)
2. One-on-one Q&A (~ 45 mins)

10 (6/10)	Recursion	Algorithms	Consolidation Lecture	Week 10 tutorial sheet Week 10 tutorial solutions	• Ed worksheets 14 and 15 due (6/10 at 6 pm)

Ed worksheets 14-15 due (6/Oct, Monday at 6 pm)
Project 2 (15%) due (17/Oct, Friday at 6 pm)

Revision: Files

type <str>
"r": read mode
"w": write mode
"a": append mode

file pointer

```
fp = open(file_name, mode)
```

type <str>
e.g. "story.txt", "data.csv"

Revision: Files modes

- "**r**"ead mode
  - Assumes that the file exists, otherwise returns a FileNotFound error.
- "**w**"rite mode
  - If file exists, it **erases** all its contents (even before writing to it!).
  - If file does not exist, it implicitly creates a new file first.
- "**a**"ppend mode
  - If file exists, writing to it will append to the end of the file.
  - If file does not exist, it implicitly creates a new file first.

write mode

append mode

Revision: Files methods (read)

- fp = open("hello.txt", "r")
- print(fp.**read()**)
- **"Hello, World!\nHow are you?\nI'm good!"**

- fp.**close()**

hello.txt

```
Hello, World!
How are you?
I'm good!
```

Revision: Files methods (readline)

- fp = open("hello.txt", "r")
- print(fp.**readline()**)
- **"Hello, World!"**
- print(fp.**readline()**)
- **"How are you?"**
- fp.**close()**

hello.txt

```
Hello, World!
How are you?
I'm good!
```

Revision: Files methods (readlines)

- fp = open("hello.txt", "r")
- print(fp.**readlines()**)
- **["Hello, World!",**
  "How are you?",
  "I'm good!"]
- fp.**close()**

hello.txt

```
Hello, World!
How are you?
I'm good!
```

Revision: Exceptions

```
try:
    # code block where an exception might occur
except ExceptionType:
    # code block to handle the exception
finally:
    # code block that will always execute, regardless of
    # whether an exception was raised or not
```

sem1-2025 > week-10 > exception.py > ...

```
1   def divide(a, b):
2       try:
3           return a / b
4       except ZeroDivisionError:
5           print("Error: Division by zero is not allowed.")
6           return None
7       except TypeError:
8           print("Error: Invalid input type. Please provide numbers.")
9           return None
10
```

Revision: CSV Files Methods (reader)

```
Victoria's Regions,2004,2005,2006,2007
Gippsland,63354,47083,51517,54872
Goldfields,42625,36358,30358,36486
Grampians,64092,41773,29102,38058
Great Ocean Road,185456,153925,150268,167458
Melbourne,1236417,1263118,1357800,1377291
```

Run PYTHON

```
1 import csv
2 visitors = open("/course/lesson15/vic_visitors.csv")
3 data = csv.reader(visitors)
4 print(list(data))
```

```
[["Victoria's Regions", '2004', '2005', '2006', '2007'], ['Gippsland', '63354', '47083'
, '51517', '54872'], ['Goldfields', '42625', '36358', '30358', '36486'], ['Grampians',
'64092', '41773', '29102', '38058'], ['Great Ocean Road', '185456', '153925', '150268',
 '167458'], ['Melbourne', '1236417', '1263118', '1357800', '1377291']]
```

✓ Program exited with code 0

Revision: CSV Files Methods (DictReader)

sem1-2025 > week-10 > data.csv > data

```
1  name,dob,age
2  Amy,1990-01-01,33
3  Bob,1995-05-15,28
4  Charlie,1988-07-20,35
```

sem1-2025 > week-10 > csv_dict_reader.py > ...

```
1  import csv
2  with open('data.csv', 'r') as fp:
3      my_reader = csv.DictReader(fp)
4      print(list(my_reader))
5
```

```
PS C:\Users\cleme\Desktop\comp10001\sem1-2025\week-10> Python csv_dict_reader.py
[{'name': 'Amy', 'dob': '1990-01-01', 'age': '33'}, {'name': 'Bob', 'dob': '1995-05-15', 'age': '28'}, {'name': 'Charlie', 'dob': '1988-07-20', 'age': '35'}]
```

Revision: Binary and Octal

- Binary
  - Consists of 0's and 1's
  - Form is 2^x, where x is the **position (from 0) from the right**.
  - e.g. $\mathbf{1011_2 = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 8 + 0 + 2 + 1 = 11_{10}}$
- Octal
  - Similar to the decimal system, but with **base 8**.
  - e.g. For the decimal $\mathbf{74_{10}}$,
    - Decimal: $\mathbf{74_{10} = 70 + 4 = 7 \times 10^1 + 4 \times 10^0}$
    - Octal: $\mathbf{74_{10} = 64 + 8 + 2 = 1 \times 8^2 + 1 \times 8^1 + 2 \times 8^0 \Rightarrow 112_8}$
  - e.g. For the binary **11111001**,
    - Make sure the binary is in **multiples of 3**, otherwise modify it => $\mathbf{011111001_2}$
    - **Divide** the binary into **lengths of 3**, then **evaluate the decimal**:
      - $\mathbf{011_2}$ in decimal is $\mathbf{3_{10}}$
      - $\mathbf{111_2}$ in decimal is $\mathbf{7_{10}}$
      - $\mathbf{001_2}$ in decimal is $\mathbf{1_{10}}$
      - Therefore, final answer is $\mathbf{371_8}$

Revision: Hexadecimal

- Similar to the decimal and octal system, but with **base 16**.
- e.g. For the decimal $\mathbf{78_{10}}$,
  - In binary this is $2^6 + 2^3 + 2^2 + 2^1 => 01001110_2$
  - Splitting into **lengths of 4**:
    - $\mathbf{0100_2}$ -> $\mathbf{4_{10}}$ -> $\mathbf{4_{16}}$
    - $\mathbf{1110_2}$ -> $\mathbf{14_{10}}$ -> $\mathbf{E_{16}}$
    - So, $\mathbf{78_{10}}$ == $\mathbf{4E_{16}}$
- Order of converting:
  - **Octal** <-> **Binary** <-> **Hexadecimal**
  - **Octal** <-> **Decimal** <-> **Binary** <-> **Hexadecimal**

Decimal to Hexadecimal Table

MATH MONKS

Decimal (Base 10)	Hexadecimal (Base 16)	Binary (Base 2)
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

Revision: Converting between bases in Python

- Decimal <-> Binary
  - **bin()** e.g. **bin(7)** returns **"0b111"** (equivalent to**"0b0111"**)
  - **int(num, base)** e.g. **int("1011", 2)** returns **7**
  - **int(0b0111)** returns **7**
- Decimal <-> Octal
  - **oct()** e.g. **oct(63)** returns **"0o77"**
  - **int(num, base)** e.g. **int("77", 8)** returns **63**
  - **int(0o77)** returns **63**
- Decimal <-> Hexadecimal
  - **hex()** e.g. **hex(230)** returns **"0xe6"** (equivalent to **"0xE6"**)
  - **int(num, base)** e.g. int**("E6", 16)** returns **230**
  - **int(0xE6)** returns **230**

TuteSheet W10 – Exercises Q1

1. Let's start by exploring how to work with files in the first question of this worksheet. There are three steps in reading and writing files:

   - Open a file: we use the [redacted] function, which takes two arguments: the file's filename as a `str`; and another `str` representing the "mode" ([redacted] for reading; [redacted] for writing (erasing all file contents if the file exists initially); and [redacted] for appending to an already-existing file). It returns a file object.

   - Read or write: Let's say we have a file object named `my_file`. We can read it using `my_file.`[redacted] method to read a whole file, returning a string; `my_file.`[redacted] to read one line of the file, returning a string; and `my_file.`[redacted] to read an entire file, returning a list with each row of the file split as a separate element in the list. Alternatively, you can simply use `for line in my_file:` to directly iterate over lines. If you want to write into `my_file` instead, use `my_file.`[redacted] method to write the `content` string into the file.

   - Close the file: close `my_file` with the `my_file.`[redacted] method to prevent buffer errors.

TuteSheet W10 – Exercises Q1

Now, fill in the blanks in the program below which reads from `in.txt` and writes to `out.txt`.

```
outfile = open("out.txt", "w")
with open("in.txt", 'r') as infile:
    line_no = 1
    for line in infile.readlines():   or simply infile:
        outfile.write(f"line: {line_no}, length: {len(line)}\n")
        line_no += 1
outfile.write("The End")
outfile.close()
```

*When you **close** a file, two things happen:*

*(1) all data that has been written/appended to the file is **"flushed"** through to the file, and it is closed on the computer's file system; and then*

*(2) the file object associated with the file **can no longer be used to manipulate the file**.*

It prevents further access to the content of the file until it is opened again.

Ed Worksheet 15: File IO and CSV Files > Files

TuteSheet W10 – Problems Q1

1. Let's have a look at exception handling. Here's the basic structure:

```
try:
    # code block where an exception might occur
except ExceptionType:
    # code block to handle the exception
finally:
    # code block that will always execute, regardless of
    # whether an exception was raised or not
```

Now, write a function `sum_and_divide_x(seq, x)` that returns the sum of `seq` divided by `x`.

This may sound like a simple task but in this case the inputs' types and values can't be guaranteed. Therefore, your function should print `"Wrong type, can't sum"` or `"Can't divide by 0"` if the corresponding issue occurs (then return None). At the end, your function should print `"Done"` to signal that it finished its calculation attempt. Here's the example function calls:

TuteSheet W10 – Problems Q1

```
>>> res1 = sum_and_divide_x([1,2,3], 2)
Done
>>> res1
3.0
```

- Returns the sum of seq divided by x.
*- At the end, should print **"Done"** to signal that it finished its calculation attempt.*

```
>>> res2 = sum_and_divide_x([1,2,"hi"], 2)
Wrong type, can't sum
Done
>>> type(res2)
<class 'NoneType'>
```

*- should print **"Wrong type, can't sum"** if inputs' types are wrong.*
- Returns None.

TypeError

```
>>> res3 = sum_and_divide_x([1,2,3], 0)
Can't divide by 0
Done
>>> type(res3)
<class 'NoneType'>
```

*- should print **""Can't divide by 0"** if a return value can't be produced because x is 0.*
- Returns None.

ZeroDivisionError

TuteSheet W10 – Problems Q1

Exception	Description
`IndexError`	Raised when a sequence index is out of range, e.g. `[][0]`.
`ZeroDivisionError`	Raised when the second argument of a division or modulo operation is zero, e.g. `1/0`.
`TypeError`	Raised when an operation or function is applied to an inappropriate type, e.g. `'e'+3`.
`NameError`	Raised when you try to reference a variable which hasn't been assigned, e.g. `num`.
`KeyError`	Raised when a key does not exist in the dictionary, e.g. `{}['a']`.

`TypeError` *and* ***`NameError`***
- issues in the code itself
- correctly-written code should never occur.

`IndexError`, `ZeroDivisionError`*, and* ***`KeyError`***

- *cannot be detected until the code is run*
- *either indicate a bug in your code or an exceptional case that needs to be handled.*

Ed Worksheet 16 > Exception Handling > Built-in Python Exceptions
https://docs.python.org/3/library/exceptions.html#bltin-exceptions
https://www.geeksforgeeks.org/built-exceptions-python/

TuteSheet W10 – Problems Q1

Now, write a function `sum_and_divide_x(seq, x)` that returns the sum of `seq` divided by `x`.

This may sound like a simple task but in this case the inputs' types and values can't be guaranteed. Therefore, your function should print `"Wrong type, can't sum"` or `"Can't divide by 0"` if the corresponding issue occurs (then return None). At the end, your function should print `"Done"` to signal that it finished its calculation attempt. Here's the example function calls:

```
def sum_and_divide_x(seq, x):
    try:
        return sum(seq) / x
    except TypeError:
        print("Wrong type, can't sum!")
    except ZeroDivisionError:
        print("Can't divide by 0!")
    finally:
        print("Done")
```

TuteSheet W10 – Problems Q2

2. When handling csv files, there are a couple of ways we can get the data out of the csv file and into our program: `csv.reader` and `csv.DictReader`. Try to use `csv.DictReader` to solve this problem.

*you have to **hard-code** which column is which by its column index, or*
***save the header row as a list** and double-check which row is which via the labels in list.*

more convenient and direct way of accessing the individual elements
*- **each row** is returned as a **dictionary**, and a **value** can be **referenced directly** by its column label*

```
import csv
visitors = open("/course/lesson15/vic_visitors.csv")
for line in csv.reader(visitors):
    print(line)
```

```
["Victoria's Regions", '2004', '2005', '2006', '2007']
['Gippsland', '63354', '47083', '51517', '54872']
['Goldfields', '42625', '36358', '30358', '36486']
```

	Victoria's Regions	2004	2005	2006	2007
0	Gippsland	63354	47083	51517	54872
1	Goldfields	42625	36358	30358	36486
2	Grampians	64092	41773	29102	38058
3	Great Ocean Road	185456	153925	150268	167458
4	Melbourne	1236417	1263118	1357800	1377291

```
import csv
visitors_dic = open("/course/lesson15/vic_visitors.csv")
for row in csv.DictReader(visitors_dic):
    print(row)
```

```
{"Victoria's Regions": 'Gippsland', '2004': '63354', '2005': '47083', '
{"Victoria's Regions": 'Goldfields', '2004': '42625', '2005': '36358',
{"Victoria's Regions": 'Grampians', '2004': '64092', '2005': '41773', '
{"Victoria's Regions": 'Great Ocean Road', '2004': '185456', '2005': '1
{"Victoria's Regions": 'Melbourne', '2004': '1236417', '2005': '126311
```

Ed Worksheet 15 > CSV Files > The csv Library
Ed Worksheet 15 > CSV Files > Naming the Elements of a Row

TuteSheet W10 – Problems Q2

Write a function `count_sales(csv_filename)`, that takes a string csv filename, and returns a dictionary that counts the frequency of products sold. On the example file shown below, it should return `{'Toy Car': 2, 'Comic Book': 1}`.

```
Date,Product,Customer
2024-03-21,Toy Car,Bluey
2024-04-12,Comic Book,Bingo
2024-05-07,Toy Car,Rusty
```

Try to use ***`csv.DictReader`***

(option) generate a csv file

```
import csv

data = [["Date", "Product", "Customer"],
        ["2024-03-21", "Toy Car", "Bluey"],
        ["2024-04-12", "Comic Book", "Bingo"],
        ["2024-05-07", "Toy Car", "Rusty"]]

filename = "sales_data.csv"
with open(filename, mode='w') as file:
    writer = csv.writer(file)
    writer.writerows(data)
print("done")
```

Assume that the csv library is already loaded

```
def count_sales(csv_filename):
    [redacted]

    with open(csv_filename, [redacted]) as file:
        for row in [redacted]:
            product = [redacted]

            if product in product_count:
                [redacted] += 1
            else:
                [redacted] = 1
    return product_count
```

TuteSheet W10 – Problems Q3(a)

Let's talk about computational counting! Binary (base 2), decimal (base 10), octal (base 8), and hexadecimal (base 16) are four commonly used number systems. The "base" is simply the number of digits we have for a specific number system. For example, the binary system has two digits (0 and 1), octal has 8 (0 to 7), and hexadecimal has 16 (0 to 9 plus A to F)!

(a) Now, count from one to twenty-one using these different number systems by completing Table 1.

Dec	Bin	Oct	Hex	Dec	Bin	Oct	Hex
0	0	0	0	11	1011	13	B
1	1	1	1	12	1100	14	C
2	10	2	2	13	1101	15	D
3	11	3	3	14	1110	16	E
4	100	4	4	15	1111	17	F
5	101	5	5	16	10000	20	10
6	110	6	6	17	10001	21	11
7	111	7	7	18	10010	22	12
8	1000	10	8	19	10011	23	13
9	1001	11	9	20	10100	24	14
10	1010	12	A	21	10101	25	15

TuteSheet W10 – Problems Q3(b)

Let's talk about computational counting! Binary (base 2), decimal (base 10), octal (base 8), and hexadecimal (base 16) are four commonly used number systems. The "base" is simply the number of digits we have for a specific number system. For example, the binary system has two digits (0 and 1), octal has 8 (0 to 7), and hexadecimal has 16 (0 to 9 plus A to F)!

(b) The conversion between these systems involves powers of their respective bases.

You are given a table with two rows: Binary and Decimal. Each column of the table represents a power of 2, starting from 2^0 on the right side and going up to 2^7. Your task is to fill in the table, providing the corresponding values in decimal and binary for each power of 2.

Power	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Decimal	128	64	32	16	8	4	2	1
Binary	10000000	1000000	100000	10000	1000	100	10	1

TuteSheet W10 – Problems Q3(c)

(c) Converting a base N number to decimal is not hard: it is simply the sum of $decimalDigit * base^{position}$ for each digit. For example, $A5B$ in hexadecimal (i.e. $A5B_{16}$) is equivalent to 2651 in decimal (i.e. 2651_{10}). This is because $10 * 16^2 + 5 * 16^1 + 11 * 16^0 = 2651$.

One common use of hexadecimal numbers is in hex colour codes, where colours are represented as a combination of red, green, and blue values, each ranging from 00 to FF (or 0 to 255 in decimal). Convert the following hex codes into their decimal red, green, and blue components and then guess the colour. The first one is done for you:

For converting #FFFF00 into decimal RBG values we can see the first two digits are FF so we can do the calculation to convert this to decimal. Since hexadecimal F represents 15 in decimal, we multiply 15 by the base 16 to the power at that position: $15 * 16^1 + 15 * 16^0 = 15 * 16 + 15 * 1 = 240 + 15 = 255$. *This means that the decimal value for the red*

Hex code	Red	Green	Blue	Guess Colour
#FFFF00	255	255	0	Yellow
#000000	0	0	0	Black
#008080	0	128	128	Teal
#BD00FF	189	0	255	Pink-ish Purple

TuteSheet W10 – Problems Q3(d)

(d) How can we convert between bases in Python?

- ***`bin()`***: *binary*
- ***`oct()`***: *octal*
- ***`hex()`***: *hexadecimal*
- ***`int(num, base)`***: *Converting to a decimal number, where* ***`num`*** *is a string containing the number to be converted to decimal and* ***`base`*** *is the base to be converted from.*

- ***`int()`***: *Writing binary, octal and hexadecimal numbers as numerical literals can be achieved by putting* ***`0b`***, ***`0o`*** *and* ***`0x`*** *respectively* ***before*** *the number*

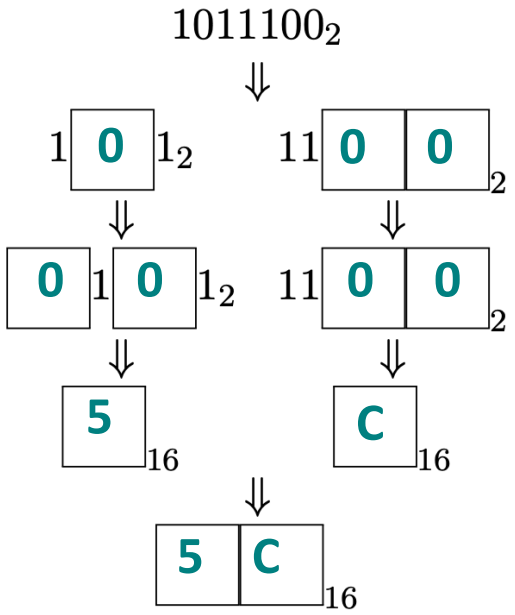
```
decimal_number = 25

# Decimal to other bases
print("Binary:", bin(decimal_number))     # '0b11001'
print("Octal:", oct(decimal_number))      # '0o31'
print("Hex:", hex(decimal_number))        # '0x19'

# Other bases to decimal
print("Binary to Decimal:", int('11001', 2))  # 25
print("Octal to Decimal:", int('31', 8))      # 25
print("Hex to Decimal:", int('19', 16))       # 25
```

```
Binary: 0b11001
Octal: 0o31
Hex: 0x19
Binary to Decimal: 25
Octal to Decimal: 25
Hex to Decimal: 25
```

TuteSheet W10 – Problems Q4(a)

Convert the following binary numbers into hexadecimal. If you're stuck, take a look at exercise 2a and think about this question: how many binary digits are required to represent a hexadecimal digit?

(a) Convert the binary number 1011100 to hexadecimal by filling in the boxes in the following diagram with a single digit:



- *Step 1: separate into 4-bit sequences*
- *Step 2: Add leading zeroes to make them all four bits long*
- *Step 3: Directly convert binary numbers (0000-1111) into hexadecimal numbers (0-F)*
- *Step 4: Combine hexadecimal numbers together, retaining place value of the original binary sequence*

TuteSheet W11 – Exercises Q5(b)

Convert the following binary numbers into hexadecimal. If you're stuck, take a look at exercise 2a and think about this question: how many binary digits are required to represent a hexadecimal digit?

(b) Convert the binary number 111101001 into hexadecimal using a method like the one shown above.

$$
\begin{array}{ccc}
 & 111101001_2 & \\
 & \Downarrow & \\
1_2 & 1110_2 & 1001_2 \\
\Downarrow & \Downarrow & \Downarrow \\
0001_2 & 1110_2 & 1001_2 \\
\Downarrow & \Downarrow & \Downarrow \\
1_{16} & E_{16} & 9_{16} \\
 & \Downarrow & \\
 & 1E9_{16} &
\end{array}
$$

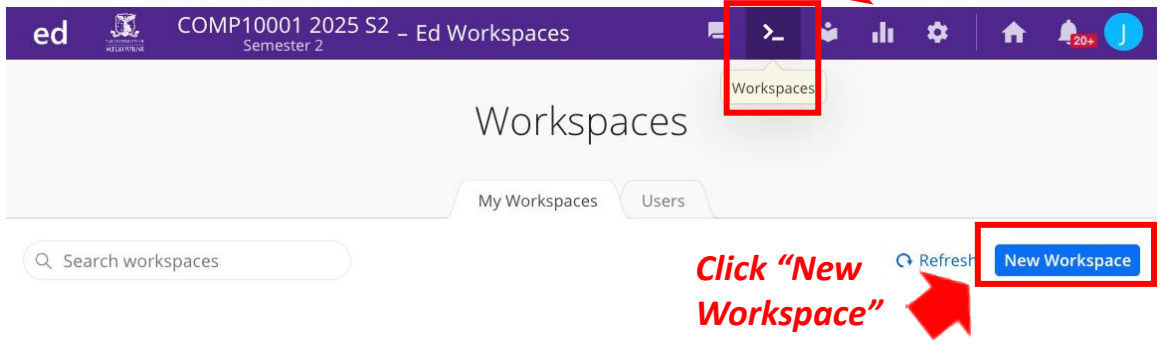
- *Step 1: separate into 4-bit sequences*
- *Step 2: Add leading zeroes to make them all four bits long*
- *Step 3: Directly convert binary numbers (0000-1111) into hexadecimal numbers (0-F)*
- *Step 4: Combine hexadecimal numbers together, retaining place value of the original binary sequence*

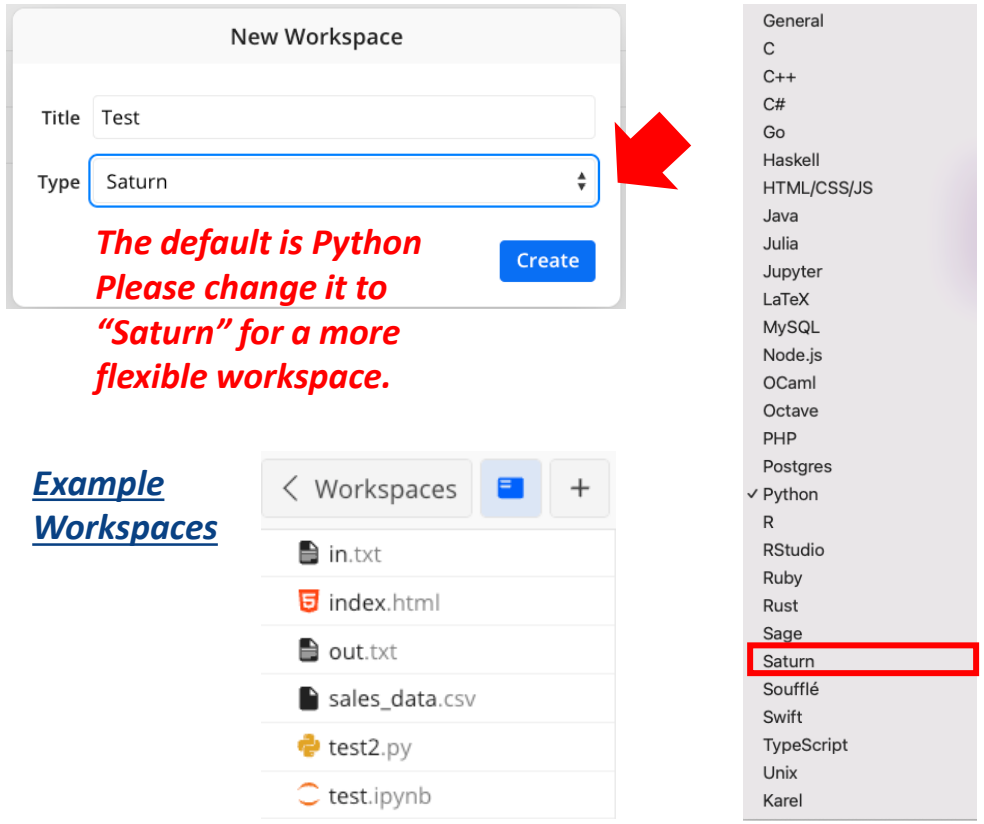
TuteSheet W10 – Revision Problems Q1-Q3

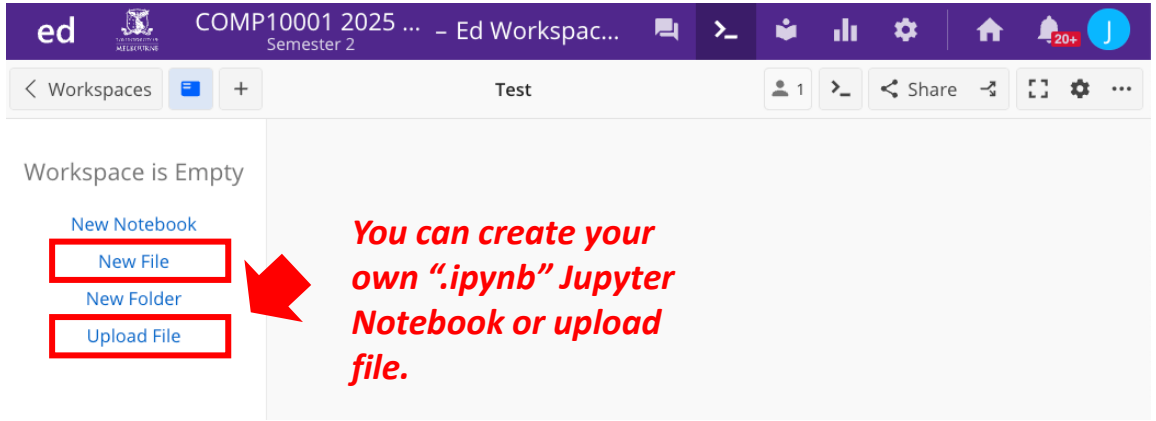
Ed Workspaces

*We can use **Ed Workspaces** for your personal programming environment.*
However, for the project, you must use each Task page only to record your development history.

How to create your workspaces?







Example Workspaces

Independent Work

- **NO Ed Worksheets due next week.**
  - Please focus on your **Project 2**. This is **due Friday, October 17th, 6pm**.
    - Project 2 is (considerably) more difficult than Project 1. **Start early**.
  - Project 1 marks and feedback should be out by now.
    - If you have any questions regarding your marks for Project 1, my email is **clement.chau@unimelb.edu.au**.
- **Raise your hand** if you have any questions!

Scan here for annotated slides