




# HW2

蔣其叡 111356024@nccu.edu.tw  
陳卉縈 112356043@nccu.edu.tw

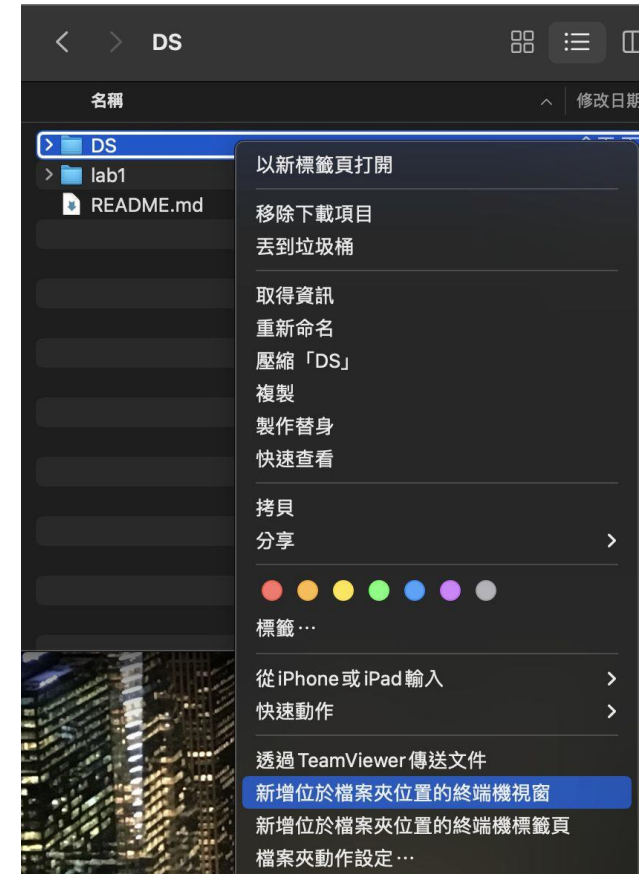
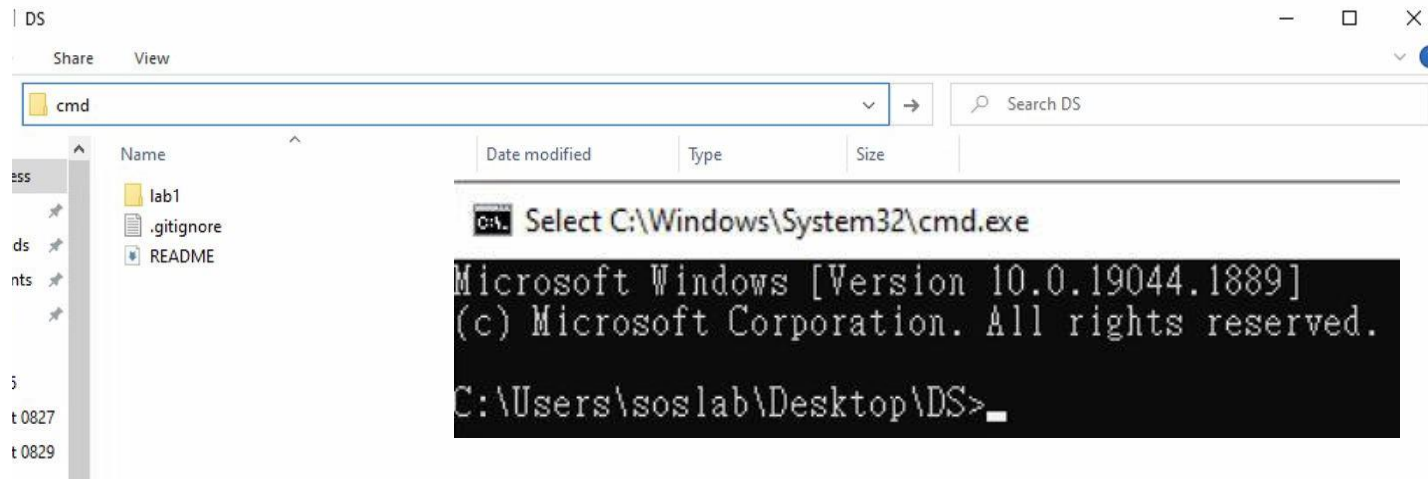


# Review - Download Homework

- In your computer's folder, use the command below :

*cd (your folder path)*

*git pull origin main*



# HW2

## Generic Geometric Progression

- Output two different types of geometric progressions using inheritance and generics
- Try to throw/catch exceptions

# HW2

## The Growth Population

- Get initial population and years from user inputs (Use Scanner class)
- Every 30 years (a generation), the population becomes double
- Output the Population Progression (by generation)
- E.g., Input: 2 people and 120 years.

Output: 2, 4, 8, 16, 32

## The Growth of Capital

- Get initial capital and years from user inputs
- The annual interest rate is 2.6%
- Output the yearly Capital progression
- E.g., Input: 100 and 2 years.

Output: 100, 102.6, 105.2676

# INPUT

- There are **three** token for inputs:
- Token 1 → type (Population or Capital):
  - 0 = Population
  - 1 = Capital
- Token 2 → number of initial people or capital = First term of geometric sequence
- Token 3 → years = how many years

# Formulation

- initial : Token 2
- rate : can be constant
- First year = initial \* rate = f1  
Second year = f1 \* rate = f2  
Third year = f2 \* rate = f3 ...

# EXAMPLE- Population

```
input → Please type (1)type and (2)number of people or initial capital and (3)years
output → 0 2 120
          2 4 8 16 32
```

# EXAMPLE - Capital

```
Please type (1)type and (2)number of people or initial capital and (3)years
input → 1 100 2
output → 100.0 102.60000000000000 1 105.26760000000000 2
```




# Exceptions

- The **try** statement allows you to define a block of code to be tested for errors while it is being executed.
- The **catch** statement allows you to define a block of code to be executed, if an error occurs in the try block.
- The **finally** statement lets you execute code, after try...catch, regardless of the result
- The **throw** statement allows you to create a custom error.

- Remember, it's **not just one input data (one line)**. You need to **stuff many input data into the program constantly**. Please use the “loop” to receive many input data and run the program.

```
Please type (1)type and (2)number of people or initial capital and (3)years
1 100 2
100.0 102.60000000000001 105.26760000000002
0 2 120
2 4 8 16 32
1 100 2
100.0 102.60000000000001 105.26760000000002
3
Error: InvalidType please enter type for 0 or 1
```

# Rules of Homework

- Project Name: HW{number of homework\_ID number} , ex: HW1\_111306XXX
- Please don't name the file like:  HW{number of homework\_ID number}.rar
- The class name where the main function of the code is located must be Main
- Remind uppercase and lowercase
- When the code is compressed and uploaded, please compress the project folder (ex: HW1\_111306XXX) into .zip(or .rar)
- Unless otherwise specified by TA, homework that cannot be compiled and executed won't be accepted.

# Rules of Homework

- Before the Lab class, we will upload sample code and slides on GitHub. Please follow the sample code we gave you to complete your homework.
- We will open the WM5 hand-in section before Lab class; the deadline usually is the midnight of the next TA class. The date may be different from the date on the teacher's slide.
- If you miss the deadline, your late homework can be made up before the end of the semester in the make-up section of WM5. But can only get 80% score for late submit.

(This make-up section will open near the end of the semester.)

# Notice

- Hand-in your HW2 via WM5
- Send Group list via Google form!

Deadline: 9/30 (Sat) 23:59

We will upload the group list before next Lab class.