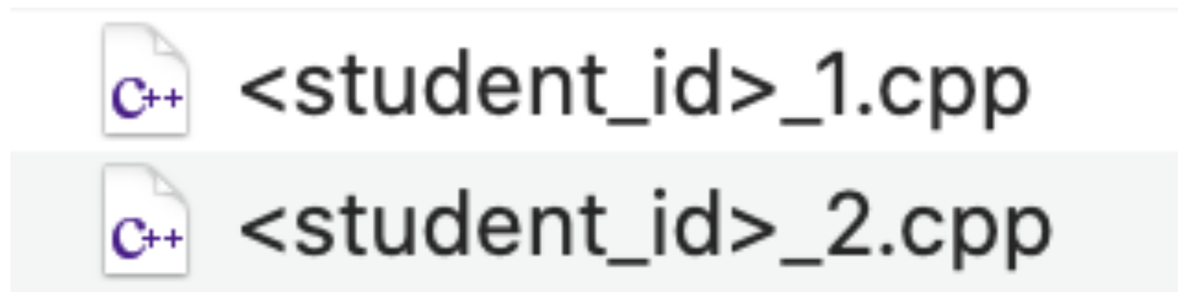
Week 4 Homework

Computer Programming Lab

2020/10/06

Remind

- 抄襲一律 0 分 （包含被抄襲者）
- 繳交期限: 10/11(Sun.) 11:59 p.m.
- 繳交的檔案格式、名稱請符合以下規定
  - 請繳交 zip檔至 Ceiba作業區，名稱為 <student_id>.zip
  - 解壓縮後須符合下圖的格式、名稱


<student_id>_1.cpp
<student_id>_2.cpp

  - e.g. b12345678.zip
- 必須完成 Demo 才可以提早離開
- 若沒有完成 Demo 就中途早退，視同缺席

Problem 1 - Determine an Armstrong number (1%)

Description

A three-digit number is an Armstrong number if it is the sum of its own digits each raised to the power of the three. For example, a three-digit number n consists of three digit a, b and c. If they satisfy the following equation:

$$a^3 + b^3 + c^3 = n$$

n is an Armstrong number. Given a three-digit number, determine whether it is an Armstrong number.

Input

A three-digit integer n, which is greater than or equal to 100 and smaller than 1000.

Output

"yes" or "no" (type:string). ("yes" if it is an Armstrong number, "no" otherwise.)

Problem 1 - Determine an Armstrong number (1%)

Sample Input

```
153
```
Plain Text

```
246
```
Plain Text

Sample Output

```
yes
```
Plain Text

```
no
```
Plain Text

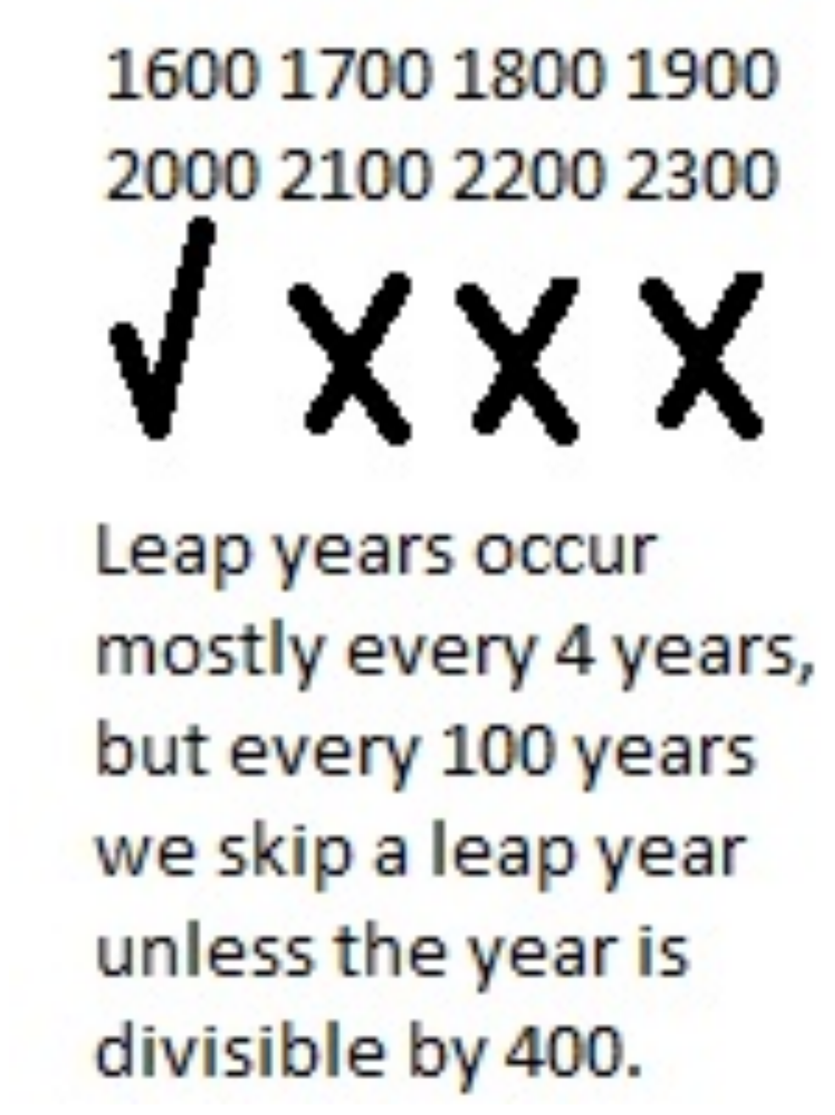
File Name

{Student_ID}_1.cpp

Problem 2 - Leap year (1%)

Description

Given a year (Common Era (CE), smaller than 3000), determine whether it is a leap year.



1600 1700 1800 1900
2000 2100 2200 2300
✓ ✗ ✗ ✗

Leap years occur
mostly every 4 years,
but every 100 years
we skip a leap year
unless the year is
divisible by 400.

Input

year (type:int).

Output

"leap year" or "common year" (type:string). ("leap year" if it is a leap year, "common year" otherwise)

Problem 2 - Leap year (1%)

Sample Input

```
2064
```

Sample Output

```
leap year
```

```
1957
```

```
common year
```

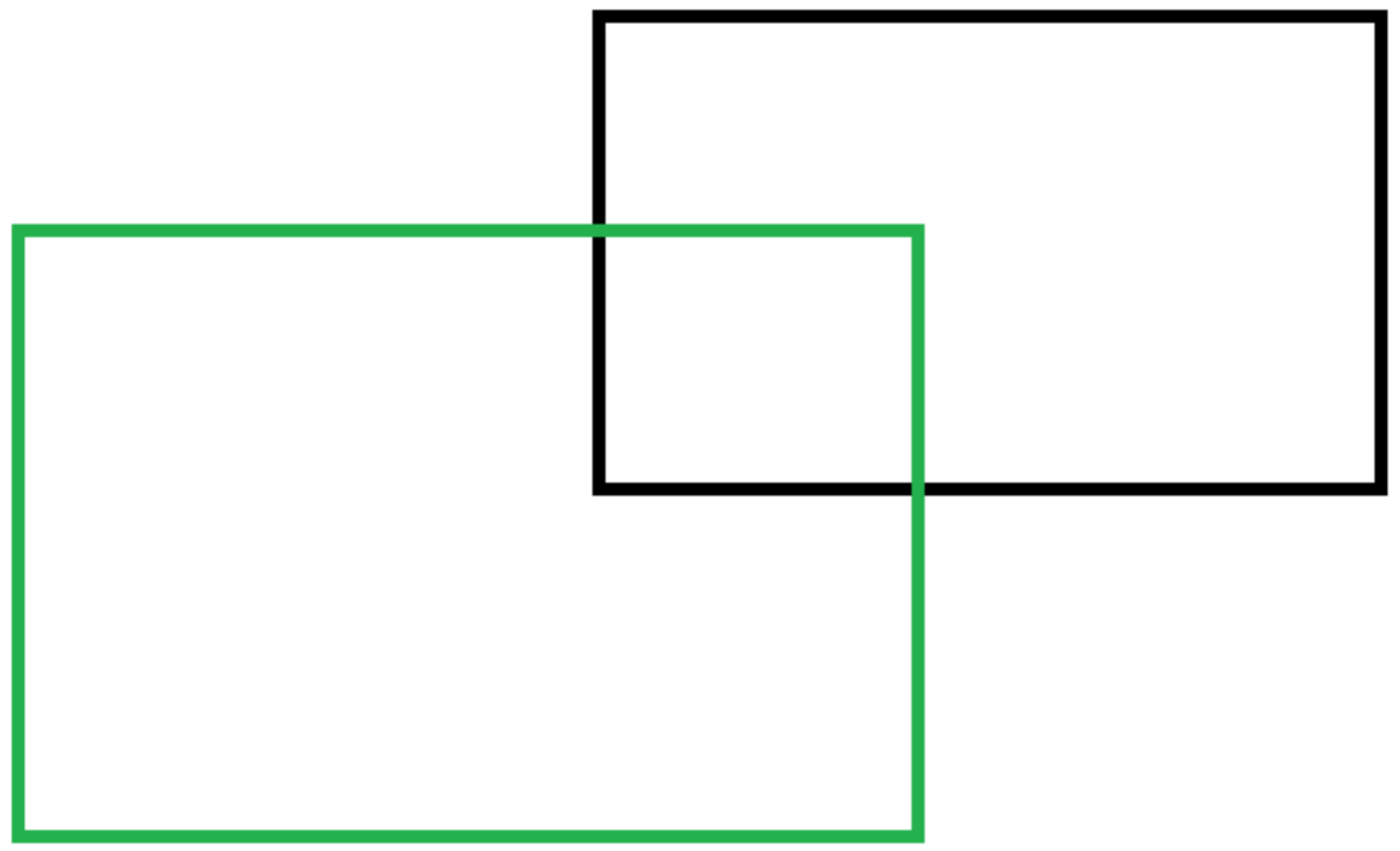
File Name

{Student_ID}_2.cpp

Problem 3 - Overlap Detection (1.5%)

Description

In this task, you will implement a program to determine if two squares overlap



Problem 3 - Overlap Detection (1.5%)

Input

User should input :

1. x1 y1 (x-y coordinate for the top left point of first rectangle)
2. x2 y2 (x-y coordinate for the bottom right point of first rectangle)
3. x3 y3 (x-y coordinate for the top left point of second rectangle)
4. x4 y4 (x-y coordinate for the bottom right point of second rectangle)

- All coordinates are pairs of integers
- All the edges of both rectangles are parallel to either x-axis or y-axis
- Overlapping does not include only touching an edge or a point.

Problem 3 - Overlap Detection (1.5%)

Output

Program should output :

"overlap" if two rectangles overlap

"no overlap" if two rectangles do not overlap

Sample Input

```
10 24
20 12
5 26
15 23
```

Plain Text

Sample Output

```
overlap
```

Plain Text

File Name

{Student_ID}_3.cpp