

# Midterm exam

Seon Joo Kim  
Yonsei University



# Outline

- Schedule
- No Plagiarism, Cheating
- Leave Comments
- Scoring
- Problem 1
- Problem 2
- Problem 3
- Problem 4
- Submission

# Deadline

- Orientation for midterm test (10:00 AM ~ 10:20 AM)
- 3 writing problems on YSCEC quiz (10:20 AM ~ 10:40 AM)
- 4 coding problems (10:40 AM ~ 12:50 PM)

---

# No Plagiarism, Cheating

- No Mercy.
- The punishment will be made to **both**.
  - the person who copied the code, and the person who shared the code.
- You can Google, **but do not copy and paste**.
- We will do plagiarism test with colleagues' codes and **codes in google**.
- Do not discuss problems with your colleagues.

---

# Leave Comments

- Leave comments in your file for TAs to understand your code.
- If no comments in the file, there may be a reduction of points.

---

# Scoring

- Total **130 points**

## **Writing problems (total 3 problems)**

Problem1 (10 points)

Problem2 (10 points)

Problem3 (10 points)

## **Coding problems (total 4 problems)**

Problem1 (10 points)

Problem2 (30 points)

Problem3 (30 points)

Problem4 (30 points)

---

# Scoring

- When ran the code, the printed text (*by cout*) in the terminal can be recorded in a file by '>>' command
- We will score the results by saving your programs printed texts by '>>', and compare by 'diff' command
- **problem1 ~ problem3** will be graded in this way.
- Example

```
$ g++ -Wall problem1.cpp -o problem1
```

```
$ ./problem1 >> output1.txt
```

```
$ diff answer.txt output1.txt
```

---

# Scoring

- **problem4** does not use input file.
- Since problem4 has multiple answers, we will not use diff.
- We will score the results by saving your programs printed texts by '>>'.  
Example

```
$ g++ -Wall problem4.cpp -o problem4
```

```
$ ./problem4 >> output4.txt
```



---

# No Additional Libraries Allowed

- Use only the given libraries in the code, and functions we gave you.

---

# Problem 1

- Print a pyramid of stars by the rule given below
- Rule:
  - The top row starts from a star at the center
  - The pyramid is of N number of lines
  - The number of stars per line increases by one with a space in between
- Input will be *given by input1.txt*
- Use problem1.cpp skeleton code
- Output the results by printing in the terminal via *cout* not by file i/o

# Problem 1

Input

```
3
1
3
7
```

Output

```
*
 *
* *
* * *
   *
  * *
 * * *
* * * *
* * * * *
* * * * *
* * * * *
```

- When ran the code, the printed text (*by cout*) in the terminal can be recorded in a file by '>>' command
- We will score the results by saving your programs printed texts by '>>', and compare by 'diff' command

## Caution

There is no space(" ") behind the last star in each line.

If space(" ") is behind the last star in each line, there will be a deduction.

```
*
 *
* *
* * *
   *
  * *
 * * *
* * * *
* * * * *
* * * * *
* * * * *
```

---

## Problem 2

- Complete a class *Student* and write two functions named “*ShowEldestStudents*”, “*ScholarshipStudentAge*”
- You need to complete class *Student* by adding public member functions. This is because data of student’s age and gpa are still hidden. But, **do not** add public variables to the class.
- Problem 2-1
  - *void ShowEldestStudents(Student s[], int sNum);*
    - argument *s* is array of *Student* class objects
    - *sNum* is number of students
  - In this function, you should print personal information of **eldest** students.
  - The output format should be like this:

```
cout <<name<<"\t"<<id<<"\t"<<age<<"\t"<<gpa<<endl;
```

---

# Problem 2

- Problem 2-2
  - *double ScholarshipStudentAge(Student s[], int sNum);*
    - argument *s* is array of *Student* class objects
    - *sNum* is number of students
  - In this function, you should return average age of students who can get scholarship.
  - Students with a gpa of 4.0 or higher are qualified for scholarship.
  - Return 0 if no students are qualified for the scholarship.
- You don't need to edit main function. All you have to do is add some member functions to class *Student* and complete the two functions.

---

## Problem 2

- input file will be given by *input2.txt*
- Your answers will be **automatically printed** if all the functions are completed.
- For your interest, the input file's format is as below

➤ Num of students

Name1

Id1

Age1

Gpa1

Name2

Id2

Age2

Gpa2

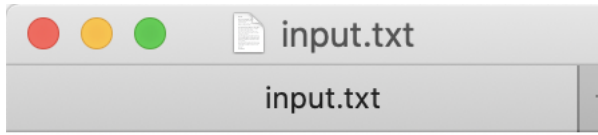
.

.

- There will be no white space in student name
- $0 < \text{Age} < 100$ ,  $0 < \text{Gpa} \leq 4.3$

## Problem 2

### Input



```
4
jennykim
2020147595
23
4.1
jiwonjung
2020147559
20
3.6
sukjunhwang
2020142632
20
4.3
yoonjuncho
2020123242
21
3.4
```

### Output

```
Average age of students who can receive scholarship : 21.50
*****
Display Eldest Students
*****
jennykim      2020147595      23      4.10
```

---

## Problem 3

- Mike wants to make a calculator dealing with two septenary numbers
  - <https://www.mathsisfun.com/definitions/septenary.html>
- With the given functions below
  - complete Septenary class' two remaining functions
    - “*sept2Dec*” and “*dec2Sept*”
  - Provide overloading of operators for two Septenary classes
    - *Addition* ( + )
    - *Subtraction* ( - )
    - *Multiplication* ( \* )
    - *Division* ( / )



---

## Problem 3

- Conversion of  $2020_7 \rightarrow 700_{10}$

To convert septenary number 2020 to decimal, follow these two steps:

Step 1) Start from one's place in 2020 : multiply ones place with  $7^0$ , tens place with  $7^1$ , hundreds place with  $7^2$  and so on from right to left

Step 2) Add all the product we got from step 1 to get the decimal equivalent of 2020.

Using the above steps, here is the work involved in the solution for converting 2020 to decimal number (Don't forget that we start from ones place to so on...)

Decimal equivalent of "0" =  $0 \times 7^0 = 0$

Decimal equivalent of "2" =  $2 \times 7^1 = 14$

Decimal equivalent of "0" =  $0 \times 7^2 = 0$

Decimal equivalent of "2" =  $2 \times 7^3 = 686$

$$- 2020_7 = 0_{10} + 14_{10} + 0_{10} + 686_{10} = 700_{10}$$

- *Please refer the link below for more examples*
  - <https://calculator.name/baseconvert/septenary/decimal/2020>

You may add functions of your own,  
but **should not** touch the functions below

## Problem 3

- You are given three functions that are helpful
- `int powerNum(int base, int exponent)`
  - Example:
    - Input : base=3, exponent=4
    - Output :  $3^4 = 81$
- `int numLength(int number)`
  - Example1:
    - Input : number=10392
    - Output : 5
  - Example2:
    - Input : number=428
    - Output : 3
- `int getPosDigit(int number, int position)`
  - Example1:
    - Input : number=10392, position=3
    - Output : 9
  - Example2:
    - Input : number=428, position=0
    - Output : 4

```
int powerNum(int base, int exponent) {
    // input (int, int)
    // output (int) : base^exponent
    // example : base=3, exponent=4 --> 3^4 = 81
    int retNum = 1;
    for(int i = 0; i < exponent; i++) {
        retNum *= base;
    }

    return retNum;
}

int numLength(int number) {
    // input (int)
    // output (int) : length of number
    // example1 : number=10392 --> 5
    // example2 : number=428 --> 3
    int length = 1;
    while(number / 10) {
        number /= 10;
        length++;
    }

    return length;
}

int getPosDigit(int number, int position) {
    // input (int, int) : position starting from 0
    // output (int) : length of number
    // example1 : number=10392, position=3 --> 9
    // example2 : number=428, position=0 --> 4
    // example2 : number=3690, position=1 --> 6
    int exponent = (numLength(number)-1) - position;
    int digit = (number / powerNum(10, exponent)) % 10;

    return digit;
}
```

---

## Problem 3

- Complete the functions below
- Problem 3-1
  - sept2Dec
    - Convert a septenary number to its corresponding decimal number
    - Input (int) : integer number in septenary
    - Output (int) : integer number in decimal
- Problem 3-2
  - dec2Sept
    - Convert a decimal number to its corresponding septenary number
    - Input (int) : integer number in decimal
    - Output (int) : integer number in septenary

---

## Problem 3

- Problem 3-3
  - Provide overloading of operators for two Septenary classes
    - *Addition ( + )*
    - *Subtraction ( - )*
    - *Multiplication ( \* )*
    - *Division ( / )*

**Hint :**

You have completed Septenary→Decimal and Decimal→Septenary

---

## Problem 3

- input file (“input3.txt”)
- Your answers will be **automatically printed** if all the problems are solved
- For your interest, the input file’s format is as below
  - N // number of test cases
  - Num1 Num2 // xN lines

Num1 and Num2 are positive integers

Num  $\geq$  Num2

---

## Problem 3

Input

```
3
5 4
25 15
66 10
```

Output

```
### Problem 1 ###
5 in decimal: 5
4 in decimal: 4
Num1 + Num2: 12
Num1 - Num2: 1
Num1 * Num2: 26
Num1 / Num2: 1

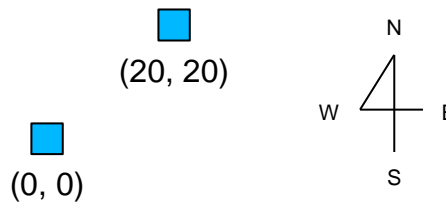
### Problem 2 ###
25 in decimal: 19
15 in decimal: 12
Num1 + Num2: 43
Num1 - Num2: 10
Num1 * Num2: 444
Num1 / Num2: 1

### Problem 3 ###
66 in decimal: 48
10 in decimal: 7
Num1 + Num2: 106
Num1 - Num2: 56
Num1 * Num2: 660
Num1 / Num2: 6
```

---

## Problem 4

- Define a class *Robot* for robots with the following properties.
- The location of a robot is given by two integer coordinates: x for its east-west position and y for its north south position (x increases as the robot moves east and y increases as the robot moves north).



- The robot can face in any of four directions (north=0, south=1, east=2, west=3).
- Write declaration and definition of the class *Robot* as well as a main function.
- Also, let the user create a robot at a particular position and move it to a particular position.

---

# Problem 4

- You must use **only** these **7 functions** in problem 4.
- You should implement **these 6 public functions** for a **class Robot**.

The 6 public functions for a class Robot is as follows:

// moves the robot a given distance in the direction it is facing; this is done by adding the distance from the appropriate coordinate.

1) void *move*(int distance);

// turn the robot to the left or right by 90 degrees.

2) void *left\_face*( );

3) void *right\_face*( );

// The get\_ functions return appropriate values: x, y, and directions respectively.

4) int *get\_x\_position*( );

5) int *get\_y\_position*( );

6) int *get\_orientation*( );

- You should implement void *show\_Robot*() to print current position in format (m\_posX, m\_posY, m\_dir). **And this function is not in Robot class.**

Example

(north=0, south=1, east=2, west=3)

When current position is (0, 0, west). Function void *show\_Robot*() print "0 0 3".



---

# Problem 4

- You should use **constructor** when creating a new robot.
- robot's default position is **(0, 0, north)**. And set this position in **default constructor**.
- Since we create two robots, you should implement **two constructors** for class Robot.
- There are **four sub-problems** in problem 4.

**Problem 4-1.** Create a **robot1** at default position **(0, 0, north)**. Then, print current position.

**Problem 4-2.** Create a **robot2** at position **(20, 20, north)**. Then, print current position.

**Problem 4-3.** Move **robot1** to position **(10, 10, east)** using only these 3 functions.

1) void *move*(int distance), 2) void *left\_face*(), 3) void *right\_face*()

And whenever the direction or position is changed, print changed position each time.

**Problem 4-4.** Move **robot2** to position **(10, 10, east)** using only these 3 functions.

1) void *move*(int distance), 2) void *left\_face*(), 3) void *right\_face*()

And whenever the direction or position is changed, print changed position each time.

# Problem 4

## Example

Q1. Create a **robot3** at default position (0, 0, west). Then, print current position.

Output

(north=0, south=1, east=2, west=3)

0 0 3

Q2. Move **robot3** to position (10, 10, north) using only these 3 functions.

1) void *move()*, 2) void *left\_face()*, 3) void *right\_face()*

And whenever the direction or position is changed, print changed position each time.

Output

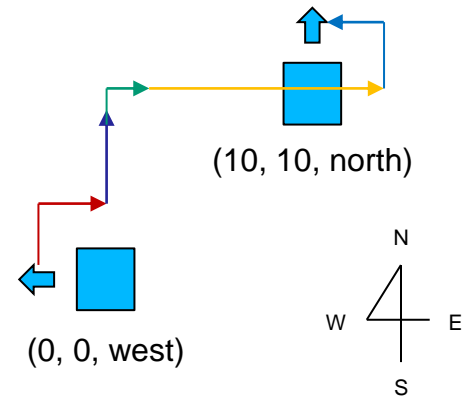
0 0 0 → turn right

0 10 0 → move 10

0 10 2 → turn right

10 10 2 → move 10

10 10 0 → turn left



---

# Problem 4

## Caution

- You must only use 7 functions mentioned before and implement 2 constructors.
- Do not simply print the answer, we will see your codes.

Example

```
cout << "0 0 0" << endl; (X)
```

- **Do not modify given skeleton code.** Only fill your codes in

```
/*
```

```
    Your Code
```

```
*/
```

---

# Submission

- Zip the folder by following steps correctly

```
oop@oop-VirtualBox:~/Desktop$ cd 2020123456_midterm/  
oop@oop-VirtualBox:~/Desktop/2020123456_midterm$ ls  
problem1.cpp  problem2.cpp  problem3.cpp  problem4.cpp  
oop@oop-VirtualBox:~/Desktop/2020123456_midterm$ cd ../  
oop@oop-VirtualBox:~/Desktop$ tar -zcvf 2020123456_midterm.tar.gz 2020123456_midterm/  
2020123456_midterm/  
2020123456_midterm/problem3.cpp  
2020123456_midterm/problem4.cpp  
2020123456_midterm/problem2.cpp  
2020123456_midterm/problem1.cpp
```

- studentId\_midterm.tar.gz
  - Ex) 2020123456\_midterm.tar.gz
- There is going to be reduction of points if not following the folder hierarchy as well
- If unzipped your submission .tar.gz file should follow the folder hierarchy below

Current directory

- studentId\_midterm.tar.gz
- studentId\_midterm
  - problem1.cpp
  - problem2.cpp
  - problem3.cpp
  - problem4.cpp

---

# Questions

- Questions should be made in **private** chat to TAs on Zoom
- For efficiency, please ask questions to the suggested TAs as listed below
- Problem 1 & Problem 4 → Yoonjun Cho
- Problem 2 → Jiwon Jung
- Problem 3 → Sukjun Hwang