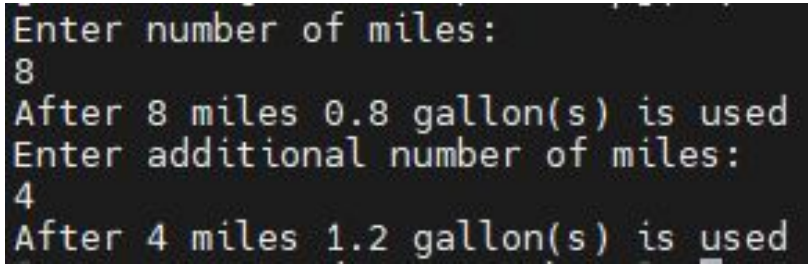
A卷-OOP Midterm

Problem A: Class (20%)

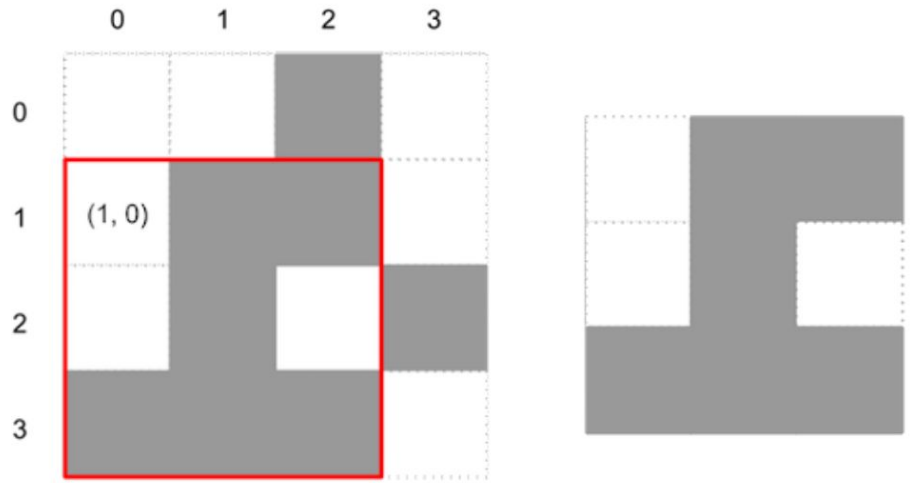
Define a class called Odometer that will be used to track fuel and mileage for an automotive vehicle. Include private member variables to track the miles driven and the fuel efficiency of the vehicle in miles per gallon. The class should have a constructor that initializes these values to zero. Include a member function to reset the odometer to zero miles, setting the fuel efficiency=10, a member function that accepts miles driven for a trip and adds it to the odometer's total, and a member function that returns the number of gallons of gasoline that the vehicle has consumed since the odometer was last reset. Use your class with a test program that creates several trips with different fuel efficiencies.

Example:



Problem B: Dynamic Array (20%)

You get one image(N*N) as the main image and several small images(M*M). M can be any number that N>M>1. Your task is to find the position where the small image is similar to a part of the main image, and output the index which is the top left.
(you don’t need to rotate the small image, and there must have one and only one fit position)



testcase:(the texts between() are not in testcase)
4(N=4) 2(2 small images)
0 0 1 0
0 1 1 0
0 1 0 1
1 1 1 0
3(M=3)
0 1 1
0 1 0
1 1 1
2(M=2)
1 0
1 0

output:
1 0
0 2

Problem C: Overloading (25%)

There are three Complex numbers in the **test.txt**. And you have **one class named Complex**, and there are **two private members** called **real and imagine**.
In this question, you are asked to **use overloading " * "** to multiply first number and second number , and then **use overloading " + "** to add the result from "*" and the third number . Finally , **using overloading " cout "** to output the number on the terminal .

For example : first number : -3+5i , second number : 4+6i , third number : 7-8i

- (-3+5i) * (4+6i) = ((-3*4) + (6*5)*-1) + ((-3*6) +(4*5)) i = -42 + 2 i
- (-42+2i) + (7-8i) = -35 - 6i
- Terminal : The answer is -35-6i

Another example : first number : 5-3i , second number : 6+7i , third number : -7+5i

- (5-3i) * (6+7i) = (5*6 + (7*-3)*-1) + ((5*7) + (-3*6)) i = 51 + 17i
- (51+17i) + (-7+5i) = 44 + 22 i
- Terminal : The answer is 44+22i

Problem D: Pointer (25%)

The information in the 5*5 maze is monster’s number. There will be a 25*2 monster information here, and the number of the monster corresponds to this 25*2 matrix seat.

You will play a player to explore in this maze. The player's initial HP is 100, the initial experience value is 0, and the initial level is 1, and the player's initial position is at (0,0).

In this program you need to read the file, in the file will give you 5*5 maze information, and give you 25*2 monster attack power and experience value, and will give you move instructions, U stands for up, D stands for down, L stands for left, R stands for right.

To defeat a monster, the player's HP will be deducted from the monster's attack power, and the experience value will be added to the experience value given by the monster. When the player's experience value reaches 100, the player will be upgraded. When the player is upgraded, the blood volume will return to 100. The experience value will return to zero, and output at each step, the output format is as follows.

(If the monster's ATK is greater than the player's HP, output "DEAD" and end the program)

(There is no need to calculate the position (0,0) at the beginning)

(The movement command will not exceed the maze, so there is no need to judge that it is beyond the boundary.)

Input format:

```
11 19 2 3 21
14 10 7 24 9
6 0 12 13 5
15 16 23 18 1
20 4 22 17 8
16 13
24 15
13 17
15 23
19 17
24 15
15 23
15 23
24 15
15 23
23 10
15 23
15 23
12 14
15 23
15 23
15 23
15 23
15 23
15 23
15 23
15 23
15 23
15 23
15 23
D R D R D R U U L U R R D D D D L L L
```

5*5 maze

25*2 monster information

Left: monster attack power

Right: experience value

move instructions

output format:

```
step 1:
level:0 hp:85 exp:23

step 2:
level:0 hp:62 exp:33

step 3:
level:0 hp:46 exp:46

step 4:
level:0 hp:31 exp:69

step 5:
level:0 hp:16 exp:92

step 6:
level:1 hp:100 exp:0

step 7:
level:1 hp:88 exp:14

step 8:
level:1 hp:73 exp:37

step 9:
level:1 hp:58 exp:60

step 10:
level:1 hp:45 exp:77

step 11:
level:1 hp:30 exp:100

step 12:
level:2 hp:100 exp:0

step 13:
level:2 hp:85 exp:23

step 14:
level:2 hp:61 exp:38

step 15:
level:2 hp:37 exp:53

step 16:
level:2 hp:13 exp:68

DEAD
```

Input:

```
11 19 2 3 21
14 10 7 24 9
6 0 12 13 5
15 16 23 18 1
20 4 22 17 8
16 13
24 15
13 17
15 23
19 17
24 15
15 23
29 23
24 15
15 23
23 10
15 23
6 7
12 14
8 12
17 25
19 6
33 16
15 23
14 7
15 23
8 12
12 15
18 23
15 9
D R D R D R U U
```

output:

```
step 1:
level:0 hp:92 exp:12

step 2:
level:0 hp:69 exp:22

step 3:
level:0 hp:53 exp:35

step 4:
level:0 hp:47 exp:42

step 5:
level:0 hp:29 exp:65

step 6:
level:0 hp:14 exp:88

step 7:
level:1 hp:100 exp:0

step 8:
level:1 hp:85 exp:9
```

Maze:

11	19	2	3	31
14	10	7	24	9
6	0	12	13	5
15	16	23	18	1
20	4	22	17	8

Position(0,0)

Problem E: Recursion (30%)

Print the permutations of size n (1<=n<=26) with recursive function.