PA Questions

Problem 1

Accept a non-zero integer as input. Print positive if it is greater than zero and negative if it is less than zero.

In [5]:
```
x = int(input())
if x > 0:
    print('positive')
else:
    print('negative')
```

```
-1
negative
```

Problem 2

Consider the piece-wise function given below.

$$f(x) = \begin{cases} x+2 & 0 < x < 10 \\ x^2+2 & 10 \leq x \\ 0 & \text{otherwise} \end{cases}$$

Accept the value of x as input and print the value of f(x) as output. Note that both the input and output are real numbers. Your code should reflect this aspect. That is, both x and f(x) should be float values.

In [9]:
```
x = float(input())
if 0 < x < 10:
    y = x + 2
elif x >= 10:
    y = x**2 + 2
else:
    y = 0.0
print(y)
```

```
0
0.0
```

Problem 3

Accept an integer as input and print the time of the day. Use the following table for reference.

Input	Output
$T < 0$	INVALID
$0 \leq T \leq 5$	NIGHT
$6 \leq T \leq 11$	MORNING
$12 \leq T \leq 17$	AFTERNOON
$18 \leq T \leq 23$	EVENING
$T \geq 24$	INVALID

The input will be a single line containing an integer. The output should be one of these strings: NIGHT, MORNING, AFTERNOON, EVENING, INVALID.

In [5]:
```
T = int(input())
if 0<=T<=5:
    print('NIGHT')
elif 6<=T<=11:
    print('MORNING')
elif 12<=T<=17:
    print('AFTERNOON')
elif 18<=T<=23:
    print('EVENING')
else:
    print('INVALID')
```

```
22
EVENING
```

Problem 4

Accept a point in 2D space as input and find the region in space that this point belongs to. A point could belong to one of the four quadrants, or it could be on one of the two axes, or it could be the origin. The input is given in 2 lines: the first line is the x-coordinate of the point while the second line is its y-coordinate. The possible outputs are first, second, third, fourth, x-axis, y-axis, and origin. Any other output will not be accepted. Note that all outputs should be in lower case.

In [11]:
```
x = float(input())
y = float(input())
if x>0 and y>0:
    print('first')
elif x<0 and y>0:
    print('second')
elif x<0 and y<0:
    print('third')
elif x>0 and y<0:
    print('fourth')
elif x==0 and y==0:
    print(origin)
elif x!=0 and y==0:
```

```
    print('x-axis')
elif x==0 and y!=0:
    print('y-axis')
```

```
0
-8
y-axis
```

Problem 5

Write a program to realize the equation of a line given 2 points (x_1, y_1) and (x_2, y_2) in 2D space. The input is in 5 lines where, the first, second, third, and fourth lines represent x_1, y_1, x_2, and y_2 respectively. The fifth line corresponds to x_3. Determine y_3 using the equation of a straight line as given below:

$$\frac{x - x_1}{x_2 - x_1} = \frac{y - y_1}{y_2 - y_1}$$

The output should be "Vertical Line" if the line is vertical. In other cases, the output should be 2 lined, where the first line is the value of y_3 and the second line indicates whether the slope of the line is positive, negative or zero. Print "Positive Slope", "Negative Slope" or "Horizontal Line" accordingly.

Note that all inputs are to be processed as real numbers.

In [13]:
```
x1, y1 = float(input()), float(input())
x2, y2 = float(input()), float(input())
x3 = float(input())

if x1 != x2:
    m =(y2-y1)/(x2-x1) # m is the slope
    y3 = y1 + m*(x3-x1)
    print(y3)

    if m == 0:
        print("Horizontal Line")
    elif m > 0:
        print("Positive Slope")
    else:
        print("Negative Slope")
else:
    print("Vertical Line")
```

```
1
4
5
6
2
4.5
Positive Slope
```

Problem 6

Accept a string as input. If the input string is of odd length, then continue with it. If the input string is of even length, make the string of odd length by following the two steps mentioned below:

(1) If the last character is a period (`.`), then remove it

(2) If the last character is not a period, then add a period (`.`) to the end of the string

Call this string of odd length `word`. Select a substring made up of three consecutive characters from `word` such that there are an equal number of characters to the left and right of this substring. Print this substring as output. You can assume that all input strings will be in lower case and will have a length of at least four.

In [15]:
```
input_string = input()
length = len(input_string)

if length % 2 == 0:
    if input_string[-1] == '.':
        input_string = input_string[:-1]
    else:
        input_string = input_string + '.'

length = len(input_string)
middle_position = (length)//2

output_string =input_string[middle_position-1:middle_position+2]
print(output_string)
```

```
abcdefghijk.
efg
```

Problem 7

A sequence of five words is called **magical** if the i^{th} word is a substring of the $(i+1)^{th}$ word for every i in the range $1 \leq i < 5$. Accept a sequence of five words as input, one word on each line. Print `magical` if the sequence is magical and `non-magical` otherwise.

Note that `str_1` is a substring of `str_2` if and only if `str_1` is present as a sequence of consecutive characters in `str_2`.

The algorithm for this problem is given as a part of the template code. Implement the algorithm in Python. This type of problem is called "Code with Algorithm".

In [18]:
```
# CODE WITH ALGORITHM
# Step-1: get the five words as input; store each word in a separate variable
word_1 = input()
word_2 = input()
word_3 = input()
word_4 = input()
word_5 = input()
# Step-2: initialise a variable count as zero
count = 0
# Step-3: for every i, if word_i is a substring of word_(i + 1), increment count
if word_1 in word_2:
    count += 1
if word_2 in word_3:
    count += 1
if word_3 in word_4:
    count += 1
if word_4 in word_5:
```

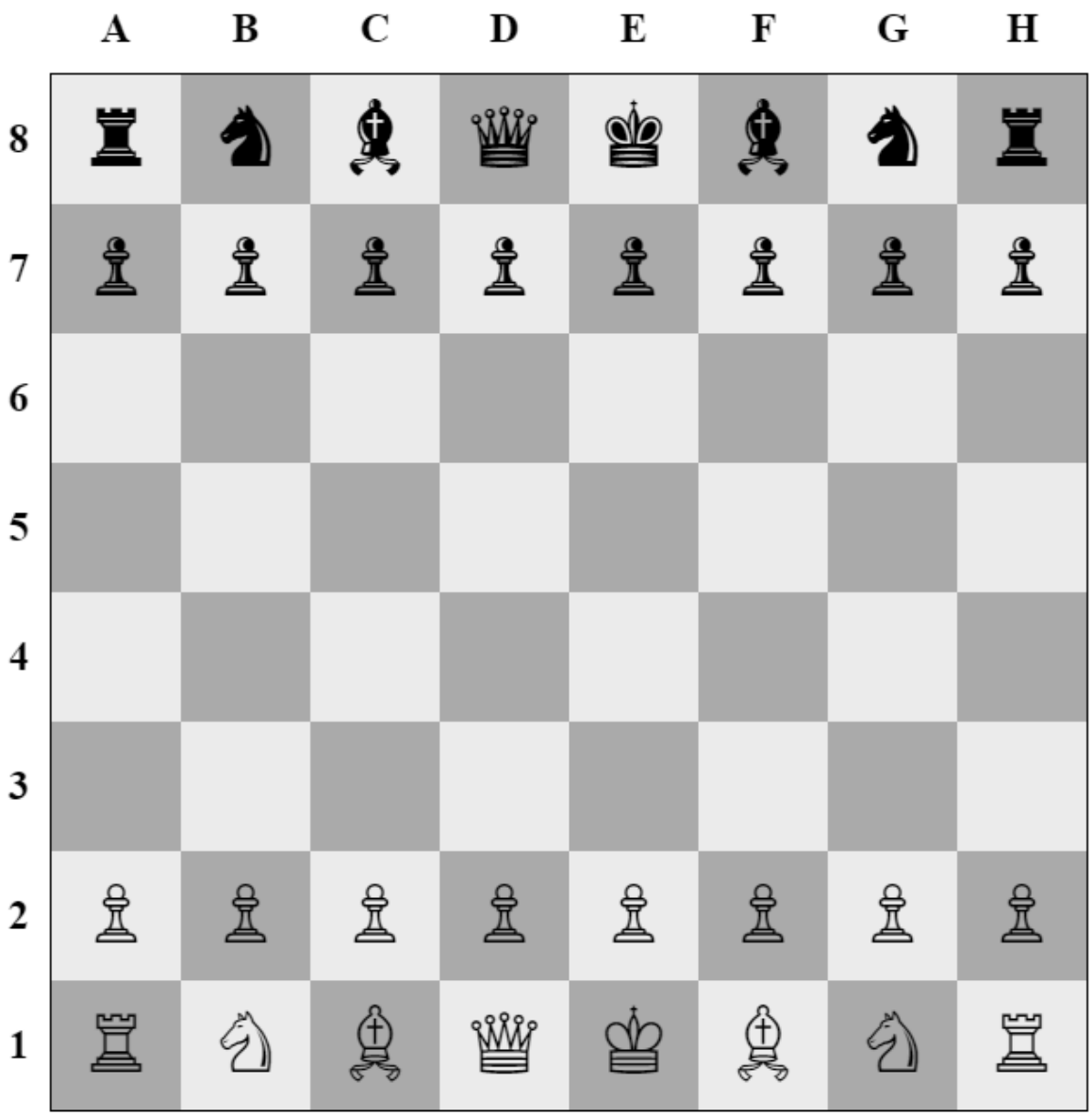
```
    count += 1

if count == 4:
    print('magical')
else:
    print('non-magical')
```

```
a
ab
abc
abcd
abcde
magical
```

Problem 8

Consider the following image of a chess-board:



Accept two positions as input: start and end. Print YES if a bishop at start can move to end in exactly one move. Print NO otherwise. Note that a bishop can only move along diagonals.

In [20]:

```
start = input()
end = input()

hor ='ABCDEFGH'
start_hor = start[0]
end_hor = end[0]

start_ver = int(start[1])
end_ver = int(end[1])

s_1 = hor.index(start_hor)
s_2 = hor.index(end_hor)
```

```
if abs(start_ver-end_ver) == abs(s_1-s_2):
    print('YES')
else:
    print('NO')
```

```
C3
E5
YES
```

Problem 9

You have n gold coins with you. You wish to divide this among three of your friends under the following conditions:

(1) All three of them should get a non-zero share.

(2) No two of them should get the same number of coins.

(3) You should not have any coins with you at the end of this sharing process.

The input has four lines. The first line contains the number of coins with you. The next three lines will have the share given to your three friends. All inputs shall be non-negative integers. If the division satisfies these conditions, then print the string `FAIR`. If not, print `UNFAIR`.

In [25]:
```
n = int(input())
share_1 = int(input())
share_2 = int(input())
share_3 = int(input())

if share_1 != 0 and share_2 != 0 and share_3 != 0:
    if share_1 != share_2 and share_2 != share_3 and share_3 != share_1:
        if share_1 + share_2 + share_3 == n:
            print('FAIR')
        else:
            print('UNFAIR')
    else:
        print('UNFAIR')
else:
    print('UNFAIR')
```

```
10
2
5
3
FAIR
```

In [27]:
```
# Alternative Method

# CODE WITH ALGORITHM
# Step 1: get the first line as an integer 'n'
# Step 2: get the next three lines as integers as share1, share2, and share3
# Step 3: initialize a boolean variable is_fair as True.
# Step 4: Check for opposites of the given three conditions one by one.
# Step 5: If the opposite of the condition is true then set is_fair to False.
# Stpe 6: if is_fair is true print "FAIR" else print "UNFAIR"

n = int(input())
share_1 = int(input())
share_2 = int(input())
share_3 = int(input())
if ((((share_1 + share_2 + share_3) == n and
```

```
    share_1 != share_2 != 0 and
    share_2 != share_3 != 0 and
    share_3 != share_1 != 0))):
    print('FAIR')
else:
    print('UNFAIR')
```

```
10
4
4
2
UNFAIR
```

Problem 10

Accept a real number x as input and print the greatest integer less than or equal to x on the first line, followed by the smallest integer greater than or equal to x on the second line.

In [29]:
```
x = float(input())
n = int(x)

if n == x:
    print(n)
    print(n)
elif x > 0:
    print(n)
    print(n+1)
else:
    print(n-1)
    print(n)
```

```
1.3
1
2
```

Problem 11

A class teacher has decided to split her entire class into four groups, namely Sapphire, Peridot, Ruby, and Emerald for sports competitions. For dividing the students into these four groups, she has followed the pattern given below:

Group							
Sapphire	1	5	9	13	17	21	...
Peridot	2	6	10	14	18	22	...
Ruby	3	7	11	15	19	23	...
Emerald	4	8	12	16	20	24	...

All the students are represented by their roll numbers. Based on the above pattern, given the roll number as input, print the group the student belongs to as output.

In [30]:
```
roll_num = int(input())
if roll_num % 4 == 1:
    print('Sapphire')
```

```
elif roll_num % 4 == 2:
    print('Peridot')
elif roll_num % 4 == 3:
    print('Ruby')
elif roll_num % 4 == 0:
    print('Emerald')
```

```
13
Sapphire
```

Problem 12

A data science company wants to hire data scientists from IIT Madras. The company follows a certain criteria for selection: for a student to be selected, the number of backlogs should be at most 5 and the CGPA (Cumulative Grade Point Average) should be greater than 6. If the student does not fit the above criteria, then the student is not offered the job. If the student is selected, then the salary offered is equal to 5 times his/her CGPA (in lakhs).

Accept the number of backlogs (integer) and the CGPA (float) of the student as input. Your task is to determine if the student is selected or not. If the student is selected, then print the package. If not, then print the string `Not Selected`.

In [31]:
```
back = int(input())
CGPA = float(input())
if back <= 5 and CGPA > 6:
    package = 5*CGPA
    print(package)
else:
    print('Not Selected')
```

```
6
7.0
Not Selected
```

Problem 13

A test match happened between Team A and Team B. The scores of the teams in both the innings are given as input in eight lines in the format given below. The first two lines represent the scores of Team A in the first innings and the next two lines represent the scores of Team A in the second innings. Likewise, the last four lines represent the scores of Team B in both the innings.

The numbers in 2nd, 4th, 6th, and 8th lines represent the wickets lost by the teams and the numbers in 1st, 3rd, 5th, and 7th represent the runs scored. For example:

120

10

210

10

115

10

189

10

In the above example, team-A has scored 120 for the loss of 10 wickets in the first innings, and 210 for the loss of 10 wickets in the second innings. Team A plays first and Team B plays second. Your task is to determine the winner of the match.

Process to decide the outcome:

Team A wins if and only if the sum of its scores in both the innings is greater than sum of the scores of Team B in both the innings AND Team B lost all the ten wickets in the second innings. Team B wins if the sum of its scores in both the innings is greater than sum of the scores of Team A in both the innings.

A match will end in a draw if the sum of scores in the two innings of both the teams are equal OR if Team B did not lose all the ten wickets in the second innings. If the match ends in a draw, then print DRAW.

Example

120

10

210

10

115

10

189

10

Example output

Team A

120 + 210 > 115 + 89 and Team B lost all 10 wickets in second innings, therefore Team A is the winner of the test match.

In [4]:
```
score_1 = int(input())
W_1 = int(input())
score_2 = int(input())
W_2 = int(input())

score_3 = int(input())
```

```
W_3 = int(input())
score_4 = int(input())
W_4 = int(input())

if score_1 + score_2 > score_3 + score_4 and W_3 + W_4 == 20:
    print('Team A')
elif score_1 + score_2 < score_3 + score_4 and W_1 + W_2 == 20:
    print('Team B')
elif score_1 + score_2 == score_3 + score_4 or W_3 + W_4 < 20 or W_1 + W_2 < 20:
    print('DRAW')
```

```
120
10
210
10
115
10
189
10
Team A
```

Problem 14

A word is said to be **perfect** if it satisfies all the following criteria:

(1) All the vowels (a,e,i,o,u) should be present in the word.

(2) Let the vowels be represented as $v_1 = a, v_2 = e, v_3 = i, v_4 = o, v_5 = u$ in lexical order.

If $i < j$, then the first appearance of v_i in the word should come before the first appearance of v_j.

If $i < j$, then the count of v_i should be greater than or equal to the count of v_j.

Accept a word as input and do the following:

(1) If it is a perfect word, print `It is a perfect word`

(2) If the word is not a perfect word, print `It is not a perfect word`

In [6]:
```
word = input()
if 'a' in word and 'e' in word and 'i' in word and 'o' in word and 'u' in word:
    if word.index('a') < word.index('e') < word.index('i') < word.index('o') < word.ir
        if word.count('a') >= word.count('e') >= word.count('i') >= word.count('o') >=
            print('It is a perfect word')
        else:
            print('It is not a perfect word')
    else:
        print('It is not a perfect word')
else:
    print('It is not a perfect word')
```

```
aeious
It is a perfect word
```

Problem 15

Accept four integers as input and write a program to print these integers in non-decreasing order.

The input will be four integers in four lines. The output should be a single line with all the integers separated by a single space in non-decreasing order.

Note: There is no space after the fourth integer.

In [8]:
```
# a b c d
a = int(input())
b = int(input())
c = int(input())
d = int(input())

if a > b:
    a, b = b, a
if a > c:
    a, c = c, a
if a > d:
    a, d = d, a
if b > c:
    b, c = c, b
if b > d:
    b, d = d, b
if c > d:
    c, d = d, c
print(a, b, c, d)
```

```
999
99
9
9999
9 99 999 9999
```

GA Questions

Problem 1

Accept three positive integers as input and check if they form the sides of a right triangle. Print YES if they form one, and NO if they do not. The input will have three lines, with one integer on each line. The output should be a single line containing one of these two strings: YES or NO.

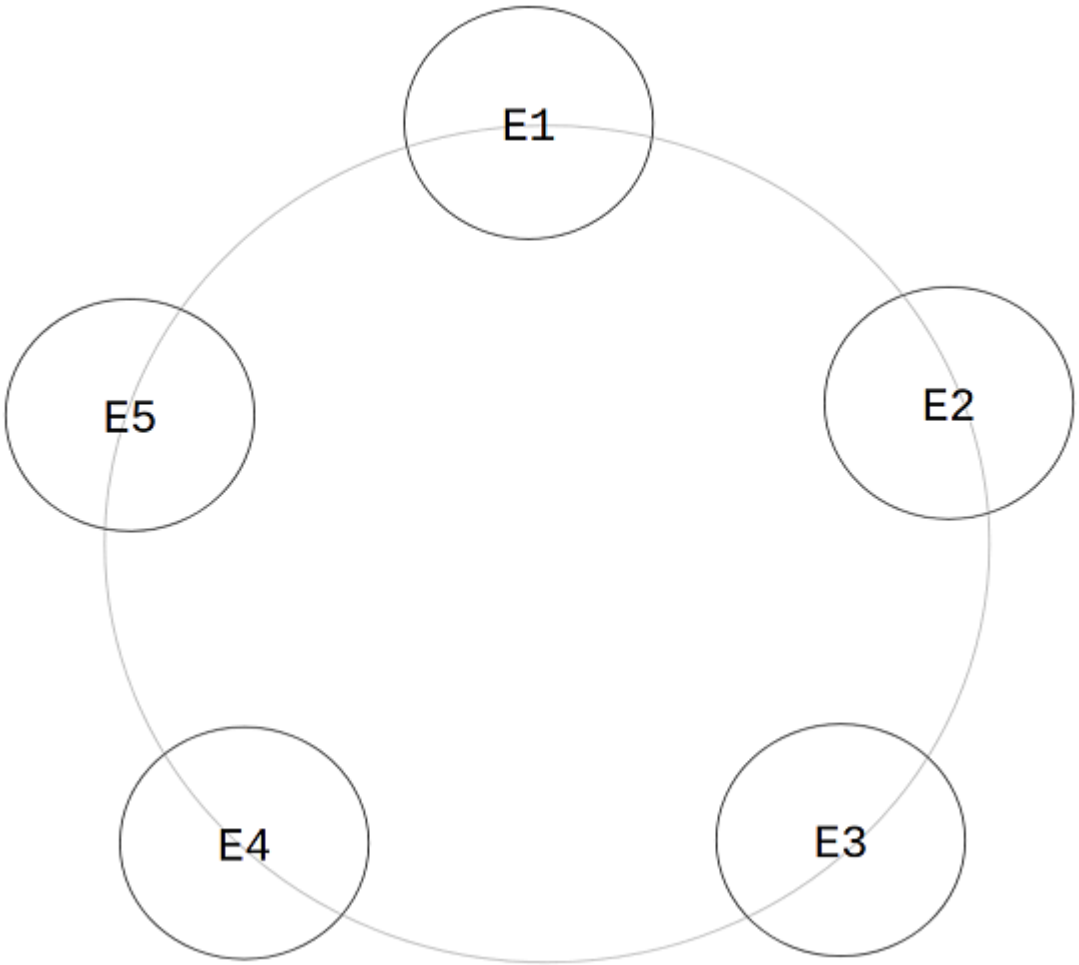
In [11]:
```
x = int(input())
y = int(input())
z = int(input())

if ((x**2 + y**2 == z**2) or
    (y**2 + z**2 == x**2) or
    (x**2 + z**2 == y**2)):
    print('YES')
else:
    print('NO')
```

```
3
4
5
YES
```

Problem 2

EvenOdd is a tech startup. Each employee at the startup is given an employee id which is a unique positive integer. On one warm Sunday evening, five employees of the company come together for a meeting and sit at a circular table:



The employees follow a strange convention. They will continue the meeting only if the following condition is satisfied.

The sum of the employee-ids of every pair of adjacent employees at the table must be an even number.

They are so lazy that they won't move around to satisfy the above condition. If the current seating plan doesn't satisfy the condition, the meeting will be canceled. You are given the employee-id of all five employees. Your task is to decide if the meeting happened or not.

The input will be five lined, each line containing an integer. The i_th line will have the employee-id of Ei. The output will be a single line containing one of these two strings: YES or NO

In [ ]:
```
e1 = int(input())
e2 = int(input())
e3 = int(input())
e4 = int(input())
```

```
e5 = int(input())

if (e1 + e2) %2 != 0:
    print('NO')
elif (e2 + e3) %2 != 0:
    print('NO')
elif (e3 + e4) %2 != 0:
    print('NO')
elif (e4 + e5) %2 != 0:
    print('NO')
elif (e5 + e1) %2 != 0:
    print('NO')
else:
    print('YES')
```

Problem 3

Accept a string as input and print the vowels present in the string in alphabetical order. If the string doesn't contain any vowels, then print the string none as output. Each vowel that appears in the input string — irrespective of its case — should appear just once in lower case in the output.

In [14]:
```
string = input()
string = string.lower()

vowels = ''
if 'a' in string:
    vowels +='a'
if 'e' in string:
    vowels +='e'
if 'i' in string:
    vowels +='i'
if 'o' in string:
    vowels +='o'
if 'u' in string:
    vowels +='u'

if len(vowels) == 0:
    print('none')
else:
    print(vowels)
```

```
Moon flowers bloom only at night, closing during the day.
aeiou
```

Problem 4

You are given the dates of birth of two persons, not necessarily from the same family. Your task is to find the younger of the two. If both of them share the same date of birth, then the younger of the two is assumed to be that person whose name comes first in alphabetical order (names will follow Python's capitalize case format).

The input will have four lines. The first two lines correspond to the first person, while the last two lines correspond to the second person. For each person, the first line corresponds to the name and the second line corresponds to the date of birth in `DD-MM-YYYY` format. Your output should be the name of the younger of the two.

In [1]:
```
c1_name = input()   # name of the first person
c1_dob = input()    # dob of the first person
c2_name = input()   # name of the second person
c2_dob = input()    # dob of the second person

# extract the day, month and year from the date of birth
# we can do this by slicing the dob string
#### person-1
c1_dob_day, c1_dob_month, c1_dob_year = int(c1_dob[: 2]), int(c1_dob[3 : 5]), int(c1_c
#### person-2
c2_dob_day, c2_dob_month, c2_dob_year = int(c2_dob[: 2]), int(c2_dob[3 : 5]), int(c2_c

younger = ''    # variable to store name of younger person
# first check for year
if c1_dob_year > c2_dob_year:
    younger = c1_name   # clearly, c1 was born after c2
elif c2_dob_year > c1_dob_year:
    younger = c2_name   # clearly, c2 was born after c1
# they share the year of birth
else:
    # check for month; same logic applies
    if c1_dob_month > c2_dob_month:
        younger = c1_name
    elif c2_dob_month > c1_dob_month:
        younger = c2_name
    # they share the month of birth; same logic applies
    else:
        # check for day
        if c1_dob_day > c2_dob_day:
            younger = c1_name
        elif c2_dob_day > c1_dob_day:
            younger = c2_name
        # if all the above conditions fail
        # then they were born on the same day
        # this is the moment we have been waiting for
        # check for alphabetical order
        else:
            if c1_name < c2_name:
                younger = c1_name   # c1 comes before c2 in alphabetical order
            else:
                younger = c2_name   # c2 comes before c1 in aplhabetical order
print(younger)
```

```
Harsh
10-03-1990
Sparsh
18-12-1987
Harsh
```

Problem 5

Accept a string as input. Your task is to determine if the input string is a valid password or not. For a string to be a valid password, it must satisfy all the conditions given below:

(1) It should have at least 8 and at most 32 characters

(2) It should start with an uppercase or lowercase letter

(3) It should not have any of these characters: / \ = ' "

(4) It should not have spaces

It could have any character that is not mentioned in the list of characters to be avoided (points 3 and 4). Output `True` if the string forms a valid password and `False` otherwise.

In [3]:
```
pw, valid = input(), False  # initialize valid to False
# first condition is being checked here
if 8 <= len(pw) <= 32:
    # second condition is being checked here
    if 'a' <= pw[0] <= 'z' or 'A' <= pw[0] <= 'Z':
        # third and four conditions are being checked here
        if not('/' in pw or '\\' in pw or '=' in pw or "'" in pw or '"' in pw or ' ' i
            valid = True
print(valid)
```

```
pass/word
False
```