Problem 1: User will input (3ages).Find the oldest one

In [16]:

```python
#solution1
def enter_age():
    age1=input("Enter age of person 1 \n")
    age2=input("Enter age of person 2 \n")
    age3=input("Enter age of person 3 \n")

    if age1>age2:
        if age1>age3:
            high=age1
        else:
            high=age3
    else:
        if age2>age3:
            high=age2
        else:
            high=age3
    print("The highest age is " + str(high))
    return()

enter_age()
```

```
Enter age of person 1
5
Enter age of person 2
6
Enter age of person 3
9
The highest age is 9
```

Out[16]:

```
()
```

Problem 1: User will input (3ages).Find the oldest one (revise this)

```
In [50]: #solution2
         for i in list(range(0,3,1)):

             age[i]=input()     #important and revise WE HAVE TO USE APPEND METHOD HERE

         print("age of oldest person:" + str(max(age[0],age[1],age[2])))
```

```
55

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[50], line 4
      1 #solution2
      2 for i in list(range(0,3,1)):
----> 4     age_[i]=input()     #important and revise
      6 print("age of oldest person:" + str(max(age[0],age[1],age[2])))

NameError: name 'age_' is not defined
```

2. Write a program that will convert celsius value to fahrenheit

```
In [33]: #(°C × 9/5) + 32 = °F
         x=input("enter temperature in deg C ")
         temp_in_F = lambda temp_in_C : temp_in_C * (9.0/5.0) + 32.0
         print(temp_in_F(float(x)))
         #print("temp_in_F:" , temp_in_F)
```

```
enter temperature in deg C 98
208.4
```

LL: lambda function and its calling data types--> convert every data type before using as formula.

3. User will input (2numbers).Write a program to swap the numbers

```
In [ ]: def swap_my_num(a,b):

            c=a
            a=b
            b=c
            return a,b
        x=input("enter the 1st number ")
        y=input("enter the 2nd number ")
        print("The swapped numbers are " + str(swap_my_num(x,y)))
```

4. Write a program that will give you the sum of 3 digits (Very important and a lot to learn)

```
In [7]: %reset -f
        number_of_elements=int(input("How many numbers do you want to enter"))
        my_num =[]
        for n in list(range(0,number_of_elements,1)):
            my_num.append(int(input()))

        print("The sum of "+str(number_of_elements) +" numbers are: " + str(sum(my_num)))
        #print("The sum of 3 numbers are:  " + sum(num[0], num[1], num[2]))
```

```
How many numbers do you want to enter2

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[7], line 5
      3 #my_num =[]
      4 for n in list(range(0,number_of_elements,1)):
----> 5     my_num.append(int(input()))
      7 print("The sum of "+str(number_of_elements) +" numbers are: " + str(sum(my_num)))

NameError: name 'my_num' is not defined
```

In [ ]:

5. Write a program that will reverse a four digit number.Also it checks whether the reverse is true.

In [ ]:

6. Write a program that will tell whether the number entered by the user is odd or even

In [10]:

```
%reset -f

for n in list(range(0,5,1)):
    num=int(input("insert your number \n"))
    if num%2==0:
        print("even")
    else:
        print("odd")
```

```
insert your number
5
odd
insert your number
6
even
insert your number
10
even
insert your number
1000
even
insert your number
0
even
```

In [ ]:

```

```

7.Write a program that will tell whether the given year is a leap year or not.

In [11]:

```
%reset -f

for n in list(range(0,5,1)):
    year=int(input("insert your year \n"))
    if year%4==0:
        if year%100==0:
            if year%400==0:
                print("leap year")
            else:
                print("non_leap year")
        else:
            print("leap year")
    else:
        print("non_leap year")
```

```
insert your year
2024
leap year
insert your year
2000
leap year
insert your year
19000
non_leap year
insert your year
1900
non_leap year
insert your year
1700
non_leap year
```

8. Write a program to find the euclidean distance between two coordinates.

In [22]:

```
%reset -f
import math
a= input("enter first x,y coordinates\n")
b= input("enter second x,y coordinates\n")

a_array=a.split(",")
b_array=b.split(",")

print("the Eucledian diatance is: " + str(math.sqrt((int(b_array[0])-int(a_array[0]))**2+(int(a_array[1])-int(b_array[1
```

```
enter first x,y coordinates
1,1
enter second x,y coordinates
2,2
the Eucledian diatance is: 1.4142135623730951
```

9. Write a program that take a user inputr of three angles and will find out whether it can form a triangle or not.

In [5]:

```
%reset -f
angles=[]
print("Enter the 3 angles in degrees")
for i in list(range(0,3,1)):
    a=input()
    angles.append(float(a))

if sum(angles)==180.0:     #ALSO THE ANGLES SHOULD NOT BE EQUALS TO ZERO
    print("it can form a triangle")
else:
    print("it can NOT form a triangle")
```

```
Enter the 3 angles in degrees
40
30
25
it can NOT form a triangle
```

In [ ]:

```

```

14.Calculate the angle between the hour hand and minute hand.

In [22]:

```
#in 1 min, minute hand moves 6 degrees and hour hand moves 0.5 degrees

%reset -f
HM_array=input("Enter time in HH:MM format(12 hours)\n").split(":")
hour_hand=int(HM_array[0])
min_hand=int(HM_array[1])
angle_hour_hand=hour_hand*60*0.5
angle_min_hand=min_hand*6

angle_dif=abs(angle_hour_hand-angle_min_hand)
if angle_dif<=180:
    print("angle difference :"+ str(angle_dif))
elif angle_dif>180:
    print("angle difference :"+ str(360-angle_dif))
else:
    print("invalid")
```

```
Enter time in HH:MM format(12 hours)
12:55
angle difference :30.0
```

15.Given two rectangles, find if the given two rectangles overlap or not. A rectangle is denoted by providing the x and y coordinates of two points: the left top corner and the right bottom corner of the rectangle. Two rectangles sharing a side are considered overlapping. (L1 and R1 are the extreme points of the first rectangle and L2 and R2 are the extreme points of the second rectangle).

In [ ]:

17. Write a program that will take three digits from the user and add the square of each digit.

In [37]:

```
%reset -f
for i in list(range(0,3,1)):
    num=int(input("enter a 3 digit number \n"))
    print(num)
    print((num%10)**2)
    print(int((num%100)/10)**2)
    print(int((num/100))**2)
    # + (num/100)**2
```

```
enter a 3 digit number 
364
364
16
36
9
```

```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
Cell In[37], line 3
      1 get_ipython().run_line_magic('reset', '-f')
      2 for i in list(range(0,3,1)):
----> 3     num=int(input("enter a 3 digit number \n"))
      4     print(num)
      5     print((num%10)**2)

File ~\anaconda3\Lib\site-packages\ipykernel\kernelbase.py:1202, in Kernel.raw_input(self, prompt)
   1200     msg = "raw_input was called, but this frontend does not support input requests."
   1201     raise StdinNotImplementedError(msg)
-> 1202 return self._input_request(
   1203     str(prompt),
   1204     self._parent_ident["shell"],
   1205     self.get_parent("shell"),
   1206     password=False,
   1207 )

File ~\anaconda3\Lib\site-packages\ipykernel\kernelbase.py:1245, in Kernel._input_request(self, prompt, ident, paren
t, password)
   1242 except KeyboardInterrupt:
   1243     # re-raise KeyboardInterrupt, to truncate traceback
   1244     msg = "Interrupted by user"
-> 1245     raise KeyboardInterrupt(msg) from None
   1246 except Exception:
   1247     self.log.warning("Invalid Message:", exc_info=True)

KeyboardInterrupt: Interrupted by user
```

```
In [ ]:
```

```
In [ ]:
```