1. Given these assignments: \( a = 5, \ b = 2, \ c = 1.5 \) and \( s = '0.5' \) write the type and value of the following expressions. Circle **ERROR** if the expression will result in a run time error.

<table>
<thead>
<tr>
<th>Statement:</th>
<th>Type:</th>
<th>Value:</th>
<th>ERROR:</th>
</tr>
</thead>
<tbody>
<tr>
<td>float(a) / b</td>
<td>float</td>
<td>2.5</td>
<td>ERROR</td>
</tr>
<tr>
<td>s * b</td>
<td>string</td>
<td>‘0.50.5’</td>
<td>ERROR</td>
</tr>
<tr>
<td>a / b</td>
<td>float</td>
<td>2.5</td>
<td>ERROR</td>
</tr>
<tr>
<td>a % b</td>
<td>int</td>
<td>1</td>
<td>ERROR</td>
</tr>
<tr>
<td>c // a</td>
<td>float</td>
<td>0.0</td>
<td>ERROR</td>
</tr>
<tr>
<td>s &gt; b</td>
<td>______</td>
<td>______</td>
<td>ERROR</td>
</tr>
<tr>
<td>s[1]</td>
<td>char</td>
<td>‘.’</td>
<td>ERROR</td>
</tr>
<tr>
<td>a // b</td>
<td>int</td>
<td>2</td>
<td>ERROR</td>
</tr>
<tr>
<td>s[0] = ‘1’</td>
<td>______</td>
<td>______</td>
<td>ERROR</td>
</tr>
</tbody>
</table>
1 Convert the following

\[(00101)_2 = (\_\_5\_\_)_{10}\]
\[(179)_{10} = (\_10110011\_)_2\]
\[(2076)_{10} = (\_81C\_\_)_{16}\]
\[(B7D)_{16} = (101101111101)_2\]

1. Given two lists, write a function to compute their intersection. Result should be a list of all the numbers appear in both input lists. The numbers in the result list can be in any order.

Sample Output:

```python
>>> num1 = [1,2,2,1]
>>> num2 = [2,2]
>>> print(intersection(num1, num2))
>>> [2]
```

```python
>>> num1 = [4,9,5]
>>> num2 = [9,4,9,8,4]
>>> print(intersection(num1, num2))
>>> [9,4] # [4,9] is also a valid answer
```

Code:

```python
def intersection(num1, num2):
    result = []
    if len(nums1) > len(nums2):
        nums1, nums2 = nums2, nums1
    dic = {}
    for i in nums1:
        if i not in dic:
            dic[i] = 1
    for i in nums2:
        if i in dic and dic[i] == 1:
            result.append(i)
    return result
```
result.append(i)

dic[i] = 0

return result

2 Given two files boxers.csv (name,nickname,strength) and matchups.csv (name1,name2):

boxers.csv

James Douglas,“Buster”,84

Rocco Marciano,"Rocky",92

Arnold Cream,"Jersey Joe Walcott”,80

Joe Frazier,“Smokin’ Joe”,95

Mike Tyson,"Iron Mike”,83

Walker Smith Jr.,“Sugar Ray Robinson”,100

Jack Dempsey,"The Manassa Mauler”,88

matchups.csv

Rocco Marciano,Jack Dempsey

Arnold Cream,Jack Dempsey

Floyd Mayweather Jr.,Rocco Marciano

Cassius Clay,Joe Frazier

Walker Smith Jr.,Floyd Mayweather Jr.

Jack Dempsey,Cassius Clay

James Douglas,Mike Tyson

Mike Tyson,Arnold Cream

Write a function to store the information from boxers.csv in a list or dictionary and return it. Write another function to read matchups.csv and the data structure you stored information before to print the winner of the matchup(The winner is the boxer with the greater strength). The main and the function prototypes are given below:

def readboxers(filename):

def boxingmatches(filename,fighter_str):

def main():

    strength_table = readboxers(‘boxers.csv’)

    boxingmatches(‘matchups.csv’, strength_table)

main()
Sample Output:

Rocco Marciano defeats Jack Dempsey
Walker Smith Jr. defeats Arnold Cream
Joe Frazier defeats James Douglas
Rocco Marciano defeats Floyd Mayweather Jr.
Cassius Clay defeats Joe Frazier
Walker Smith Jr. defeats Floyd Mayweather Jr.
Cassius Clay defeats Jack Dempsey
James Douglas defeats Mike Tyson
Mike Tyson defeats Arnold Cream

Code:

```python
def readboxers(filename):
    box_str = {}
    fh = open(filename)
    for line in fh:
        line = line.strip()
        lst = line.split(',')
        name_key = lst[0]
        str_value = int(lst[2])
        box_str[name_key] = str_value
    fh.close()
    return box_str

def boxingmatches(filename,fighter_str):
    fh = open(filename)
    for line in fh:
        line = line.strip
        arr = line.split(',,')
```
ANSWER KEY

```python
def main():
    strength_table = readboxers('boxers.csv')
    boxingmatches('matchups.csv',strength_table)
```

1. Write a function (just a function) that iterates through a list of integers and checks if any of the odd integers are in increasing order. The return type should be a boolean value. You can assume the values will be positive.

   **Sample Outputs:**
   
   `checkfuntion([1,4,2,3,8,10,5])`: True
   
   `checkfuntion([2,6,3,12,13,20,11])`: False

   **Code:**
   
   ```python
def checkfunction(lst):
    prev_odd = 0
    for num in lst:
        if (num%2 == 1 and num > prev_odd):
            prev_odd = num
        elif (num%2 == 1 and num < prev_odd):
            return False
    return True
```
3. Write a function (just the function) that will sum the numbers in the string and return the result.

Sample Output (if printed):

Enter a string: x2m56ii
The sum is: 13

Code:

```python
def sumNum(word):
    sum = 0
    for letter in word:
        if (letter >= '0' and letter <= '9'):
            sum += int(letter)
    return sum
```

1. Write a function that takes a temperature in Fahrenheit from user input, and if the number is not divisible by 2 convert it to Kelvin, if it is divisible by 2 converts it to Celsius. However, if any number is divisible by 3 do nothing to it. Return the number value and a string that indicates what is the resulting temperature system. Then write the main to prompt the user for the temperature and print out the information. Limit to 3 decimal places.

   (Hint: $K = (F + 459.67) \times \frac{5}{9}, C = (F - 32) \times \frac{5}{9}$)

Sample Output:

Enter temperature in Fahrenheit: 67
The temperature in Kelvin is 292.594

Enter temperature in Fahrenheit: 21
The temperature in Fahrenheit is 21.000

Enter temperature in Fahrenheit: 88
The temperature in Celsius is 31.111
Code: NEXT PAGE:

def convert(temp):
    if temp % 3 == 0:
        tempType = 'Fahrenheit'
        return temp,tempType
    if temp % 2 == 0:
        converted -= 32
        tempType = 'Celsius'
    else:
        converted += 459.67
        tempType = 'Kelvin'
    converted = converted * 5 / 9
    return temp,tempType

def main():
    faren = input('Enter temperature in Fahrenheit: ')
    converted_temp,type = convert(faren)
    print("The temperature in {0} is {:.3f}".format(converted_temp,type))

1. Complete the following class definition. (Note: c3 and c4 must be type "Complex"). Note that the class is initialized with a string of form “a+bi” (look at the main function).

class Complex:
    def __init__(self, s):  # Modify this
self.real =
self.img =

def add(self, other):
    # Write this

def sub(self, other):
    # Write this

def print(self):
    # Write this

def main():
    c1 = Complex("1+2i")
    c2 = Complex("-3-4i")
    c3 = c1.add(c2)
    c4 = c1.sub(c2)
    c3.print()
    c4.print()

Expected output:
>> '-2-2i'
>> '4+6i'

Code:
class Complex:
    def __init__(self, s):
        real_sign = 1
        if(s[0] == '-'):
            s = s[1:]
            real_sign = -1
elif(s[0] == '+'):
    s = s[1:]
else:
    if '+' in s:
        s = s.split('+')
        self.real = real_sign * int(s[0])
        self.img = int(s[1]):-1)
    else:
        s = s.split('-')
        self.real = real_sign * int(s[0])
        self.img = int(s[1]):-1)*-1

def add(self, other):
    real = self.real + other.real
    img = self.img + other.img
    if img < 0 :
        return Complex( str(real) + "-" + str(abs(img))+ "i" )
    else:
        return Complex( str(real) + "+" + str(img) + "i" )

def sub(self, other):
    real = self.real - other.real
    img = self.img - other.img
    if img < 0 :
        return Complex( str(real) + "-" + str(abs(img))+ "i" )
    else:
        return Complex( str(real) + "+" + str(img) + "i" )
def print(self):
    if self.img < 0 :
        return str(self.real) + "-" + str(abs(self.img)) + "i"
    else:
        return str(self.real) + "+" + str(self.img) + "i"