You can view this report online at : https://www.hackerrank.com/x/tests/1246863/candidates/33265019/report

Full Name:	Rohan Raj
Email:	rr07656@st.habib.edu.pk
Test Name:	**CS 101 - Lab 10 - Fall 2021**
Taken On:	30 Nov 2021 12:24:28 PKT
Time Taken:	9329 min 54 sec/ 14580 min
Work Experience:	< 1 years
Invited by:	Aisha
Skills Score:	
Tags Score:	CS101 230/230 Lists 230/230 NestedLists 120/120 Strings 50/50

100%
500/500

scored in **CS 101 - Lab 10 - Fall 2021** in 9329 min 54 sec on 30 Nov 2021 12:24:28 PKT

Recruiter/Team Comments:

No Comments.

	Question Description	Time Taken	Score	Status
Q1	Divisible by 7 but are not a multiple of 5 > Coding	12 min 12 sec	30/ 30	✓
Q2	Odd List > Coding	11 min 43 sec	40/ 40	✓
Q3	First and Last > Coding	7 min 40 sec	30/ 30	✓
Q4	Remove Duplicates > Coding	1 hour 5 min 34 sec	30/ 30	✓
Q5	Common > Coding	10 min 2 sec	30/ 30	✓
Q6	Breakdown > Coding	1 hour 37 min 4 sec	40/ 40	✓
Q7	Merge Lists > Coding	10 min 35 sec	30/ 30	✓
Q8	Compute My Bill > Coding	56 min 15 sec	120/ 120	✓
Q9	Count Characters > Coding	1 hour 32 min 28 sec	50/ 50	✓
Q10	Check Types > Coding	4 hour 11 min 27 sec	60/ 60	✓
Q11	Problem Solving - Pattern 4 > Coding	1 hour 22 min 32 sec	20/ 20	✓
Q12	Problem Solving - Pattern 5 > Coding	1 hour 16 min 26 sec	20/ 20	✓

QUESTION 1

Correct Answer

Score 30

Divisible by 7 but are not a multiple of 5 › Coding

QUESTION DESCRIPTION

Problem:

Write a function `seven_or_five()` which returns a list of all such numbers which are divisible by 7 but are not a multiple of 5, between `a` and `b` (both included).

Sample:

```
>>>seven_or_five(10 , 50)
[14, 21, 28, 42, 49]
```

CANDIDATE ANSWER

Language used: **Python 3**

```python
def seven_or_five(a,b):
    lst = []
    for i in range(a, b+1):
        if i % 7 == 0 and i % 5 != 0:
            lst.append(i)
    return lst
# Enter your code here. Read input from STDIN. Print output to STDOUT


```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	Success	10	0.0263 sec	8.01 KB
Testcase 1	Easy	Sample case	Success	10	0.0284 sec	7.93 KB
Testcase 2	Easy	Hidden case	Success	10	0.026 sec	8.19 KB

No Comments

QUESTION 2

Correct Answer

Score 40

Odd List › Coding

QUESTION DESCRIPTION

Problem:

Write a Python function `odd_items()` that takes a parameter `numList` of type list and returns the list containing odd numbers only.

Sample:

```
>>>odd_items([23,44,6,7,9,0]
[23,7,9]
>>>odd_items([2,4,6,3,1,8]
[3,1]
```

CANDIDATE ANSWER

Language used: **Python 3**

```
1 def odd_items(lst):
2     x = []
3     for i in lst:
4         if i % 2 == 1:
5             x.append(i)
6     return x
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	Success	10	0.0398 sec	8.91 KB
Testcase 1	Easy	Sample case	Success	10	0.0546 sec	8.93 KB
Testcase 2	Easy	Hidden case	Success	10	0.0598 sec	9.07 KB
Testcase 3	Easy	Hidden case	Success	10	0.0372 sec	8.99 KB

No Comments

QUESTION 3

Correct Answer

Score 30

First and Last › Coding

QUESTION DESCRIPTION

Problem

Write a function `first_and_last()` that takes a list and makes a new list of only the first and last elements of the given list.

Sample

```
>>>first_and_last([2,4,5,6,7,7,100])
[2,100]

>>>first_and_last(['Twinkle' , 'Twinkle' , 'little' , 'star'])
['Twinkle','star']
```

CANDIDATE ANSWER

Language used: **Python 3**

```
1 def first_and_last(lst):
2     x = []
3     c = 0
4     x.append(lst[c])
5     for i in lst:
6         c += 1
7     x.append(lst[c - 1])
8     return x
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	Success	10	0.0655 sec	9.11 KB
Testcase 1	Easy	Sample case	Success	10	0.0383 sec	9.03 KB
Testcase 2	Easy	Sample case	Success	10	0.0541 sec	9.02 KB

No Comments

QUESTION 4

Correct Answer

Score 30

Remove Duplicates › Coding

QUESTION DESCRIPTION

Problem

Write a function `removeDuplicates()` that takes a list and returns a new list that contains all the elements of the first list minus all the duplicates.

Sample

```
>>>removeDuplicates([1,1,1,3,4,5,5])
[1,3,4,5]
>>>removeDuplicates([10,30,10,35,20,33,30,50,10])
[10,30,35,20,33,50]
```

CANDIDATE ANSWER

Language used: **Python 3**

```
def removeDuplicates(lst):
    y = []
    c = 0
    p = 0
    x = 0
    for i in lst:
        c += 1
    for k in lst:
        x = 0
        for j in range (c):
            if lst [j] == k:
                if k not in y:
                    y.append(k)

    return y

```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	Success	10	0.0519 sec	8.88 KB
Testcase 1	Easy	Sample case	Success	10	0.0591 sec	9.04 KB
Testcase 2	Easy	Sample case	Success	10	0.0407 sec	8.96 KB

No Comments

QUESTION 5

Correct Answer

Score 30

Common › Coding

QUESTION DESCRIPTION

Problem

Write a function named `common` that, given two lists, returns a list containing the common elements in the

lists.

Sample

```
>>> common(['1','1','1','1','1'],['1','2','3'])
['1']
>>> common(['a', 'b', '34'],['34', '34', '34'])
['34']
>>> common(['2', '4', '6', '8', '10'],['2', '3', '5', '7', '11', '13',
'17'])
['2']
>>> common([0,1,2,3,4],[0,1,1,2,3,5,8,13])
['0', '1', '2', '3']
>>> common(['a', 'b', 'c'],['abc', 'def'])
[]
```

INTERVIEWER GUIDELINES

Solution

```
lst1 = input().strip().split()
lst2 = input().strip().split()

def common(lst1, lst2):
    common = []
    for item in lst1:
      if item not in common:
        common.extend([item] * min(lst1.count(item), lst2.count(item)))
    return common
```

CANDIDATE ANSWER

Language used: **Python 3**

```
1 def common(lst1, lst2):
2     x = []
3     for i in lst1:
4         for j in lst2:
5             if (i == j) and (j not in x):
6                 x.append(j)
7     return x
8 # Enter your code here.
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	Success	5	0.0255 sec	7.91 KB
Testcase 1	Easy	Sample case	Success	5	0.0226 sec	7.88 KB
Testcase 2	Easy	Hidden case	Success	5	0.0306 sec	7.85 KB
Testcase 3	Easy	Hidden case	Success	5	0.0368 sec	7.8 KB
Testcase 4	Easy	Hidden case	Success	5	0.0267 sec	7.9 KB
Testcase 5	Easy	Hidden case	Success	5	0.0295 sec	7.96 KB

No Comments

QUESTION 6

Correct Answer

Score 40

Breakdown › Coding

QUESTION DESCRIPTION

Problem

Write a function named 'breakdown' that takes a parameter 'n' as an integer and returns a list containing all the units, tens, thousands, etc. that make up the number.

```
>>> breakdown(1256)
[1000, 100, 100, 10, 10, 10, 10, 10, 1, 1, 1, 1, 1, 1]
>>> breakdown(-2342)
[-1000, -1000, -100, -100, -100, -10, -10, -10, -10, -1, -1]
```

Constraints

`n` contains no more than 5 digits.

INTERVIEWER GUIDELINES

Solution

```
def breakdown(n):
    breakdown = []
    # Remember if n is negative and make it positive.
    is_negative = False
    if n < 0:
        is_negative = True
        n = -n
    # Get each digit of the number by dividing it by increasing powers of
10.
    for power in range(5):
        power_of_10 = 10**power
        q, r = divmod(n, power_of_10)
        # Store power of 10. Negate if n was negative.
        if is_negative:
            power_of_10 = -power_of_10
            breakdown.extend([power_of_10]*r)
            n = q
    # Reverse to meet required order.
    breakdown.reverse()
    return breakdown
```

CANDIDATE ANSWER

Language used: **Python 3**

```
1  def breakdown(n):
2      a = 0
3      b = 0
4      c = []
5      if n < 100000 and n > -100000:
6          while n > 0:
7              if n // 10 == 0:
8                  b = 1
9                  a = n // b
10                 for i in range (a):
11                     c.append(b)
12                 n = n // 10
13             if n // 100 == 0:
```

```
                b = 10
                a = n // b
                for i in range (a):
                    c.append(b)
            elif n // 1000 == 0:
                b = 100
                a = n // b
                for i in range (a):
                    c.append(b)
            elif n // 10000 == 0:
                b = 1000
                a = n // b
                for i in range (a):
                    c.append(b)
            elif n // 100000 == 0:
                b = 10000
                a = n // b
                for i in range (a):
                    c.append(b)
            n = n % b

        while n < -1:
            if n // 10 == -1:
                b = -1
                a = n // b
                for i in range (a):
                    c.append(b)
                n = n // 10
            if n // 100 == -1:
                b = -10
                a = n // b
                for i in range (a):
                    c.append(b)
            elif n // 1000 == -1:
                b = -100
                a = n // b
                for i in range (a):
                    c.append(b)
            elif n // 10000 == -1:
                b = -1000
                a = n // b
                for i in range (a):
                    c.append(b)
            elif n // 100000 == -1:
                b = -10000
                a = n // b
                for i in range (a):
                    c.append(b)
            n = n % b
        return c
# Enter your code here.


```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Hidden case	Success	10	0.0362 sec	8.3 KB
Testcase 1	Easy	Sample case	Success	10	0.0289 sec	8.2 KB
Testcase 2	Easy	Hidden case	Success	10	0.0329 sec	8.36 KB
Testcase 3	Easy	Sample case	Success	10	0.0293 sec	8.31 KB

No Comments

QUESTION 7

Correct Answer

Score 30

Merge Lists › Coding

QUESTION DESCRIPTION

Problem

Write a function `merge_lists()` that takes two parameters `lst1` and `lst2` of type list sorted in increasing order and returns a merged list of all the elements in sorted order.

Sample

```
>>> merge_lists(['aa', 'xx', 'zz'], ['bb', 'cc'])
['aa', 'bb', 'cc', 'xx', 'zz']

>>> merge_lists(['edf', 'dds'], ['ahha', 'gfhfg', 'dds'])
['ahha', 'dds', 'dds', 'edf', 'gfhfg']
```

CANDIDATE ANSWER

Language used: **Python 3**

```
1 def merge_lists(lst1 , lst2):
2     a = []
3     a = (lst1 + lst2)
4     a.sort()
5     return (a)
6
7
8 # Enter your code here. Read input from STDIN. Print output to STDOUT
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	Success	10	0.0348 sec	9.03 KB
Testcase 1	Easy	Sample case	Success	10	0.0559 sec	9.25 KB
Testcase 2	Easy	Hidden case	Success	10	0.0411 sec	9.08 KB

No Comments

QUESTION 8

Correct Answer

Score 120

Compute My Bill › Coding Lists NestedLists CS101

QUESTION DESCRIPTION

Problem

You've got everything that you need on your shopping list. The grocery store management system needs your help in computing a bill. The IT department has generated a list containing details of all the items you have bought. However, they are having trouble computing the bill. See if you can help the IT department by using your exceptional programming skills.

using your exceptional programming skills.

Write a function '`compute_my_bill()`' that takes as parameter a list '`lst`' and returns a receipt.

Sample interaction

```
>>> lst = [ ['Milk' , 10.99 , 2],['Cake' , 12.50 , 5],['Tapal Family
Mixture' , 450 , 1],      ['Shan Bombay Biryani' , 70 , 2]]

>>> comput_my_bill(lst)

[ ['Milk', 10.99, 2, 21.98], ['Cake', 12.5, 5, 62.5], ['Tapal Family
Mixture', 450, 1, 450], ['Shan Bombay Biryani', 70, 2, 140], ['Quantity ',
10, 'Total Bill', 674.48] ]
```

Input/Output

Your function will be provided the list that contains nested lists, each of which contains ItemName, followed by Unit Price and Quantity of each item Your function should return a receipt that contains nested lists, each of which contains ItemName, Unit Price, Quantity, and Total Price of an item and one nested list that contains total items bought and the total amount that a customer has to pay.

INTERVIEWER GUIDELINES

```
def comput_my_bill(lst):
    r = []
    totalBill = 0
    qty = 0
    for i in lst:
        itemtotalamount = i[1] * i[2]
        totalBill += itemtotalamount
        r.append([i[0] , i[1], i[2] , itemtotalamount])
        qty += i[2]
    r.append( [ 'Quantity ' , qty ,'Total Bill',  totalBill])
    print(r)
```

CANDIDATE ANSWER

Language used: **Python 3**

```
def compute_my_bill(lst):
    x = []
    y = []
    q = 0
    b = 0
    c = 0
    for i in lst:
        c = (i[1] * i[2])
        i.append(c)
        q += i[2]
        b += c
        x.append(i)
    y.append("Quantity ")
    y.append(q)
    y.append("Total Bill")
    y.append(b)
    x.append(y)
    return x
print(compute_my_bill(eval(input())))

```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	Success	60	0.0265 sec	7.93 KB
Testcase 1	Easy	Hidden case	Success	60	0.0283 sec	7.82 KB

No Comments

QUESTION 9

Correct Answer

Score 50

Count Characters › Coding

Strings | Lists | CS101

QUESTION DESCRIPTION

Write a function `count_char` that takes a parameter `s` as a string and returns a list containing nested-list pairs of the count (frequency) of each character, including special characters and spaces.

```
>>> count_char("This is easier than it looks")
[['t', 3], ['h', 2], ['i', 4], ['s', 4], [' ', 5], ['e', 2], ['a', 2],
['r', 1], ['n', 1], ['l', 1], ['o', 2], ['k', 1]]
>>> count_char(["Don\'t be foolish"])
"Error: bad argument. Function 'count_char' only accepts strings."
```

INTERVIEWER GUIDELINES

Solution

```
def count_char(text):
    if not isinstance(text, str):
        return "Error: bad argument. Function 'count_char' only accepts
strings."
    unique_char = []
    count_char = []
    # Convert text to lower case, keep track of the unique characters,
    # and save their count.
    text = text.lower()
    for c in text:
        if c not in unique_char:
            unique_char.append(c)
            count_char.append([c, text.count(c)])
    return count_char
```

CANDIDATE ANSWER

Language used: **Python 3**

```
1  def count_char(s):
2      if type(s) != str:
3          return ("Error: bad argument. Function 'count_char' only accepts
4  strings.")
5      else:
6          a = s.lower()
7          y = []
8          z = []
9          h = ""
10         for i in a:
11             y = []
```

```
            y = []
            x = 0
            if i not in h:
                for j in range (len(s)):
                    if i in a[j]:
                        x += 1
                h += i
                y.append(i)
                y.append(x)
                z.append(y)
    return z
# Enter your code here.

```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	Success	10	0.0374 sec	9.09 KB
Testcase 1	Easy	Hidden case	Success	15	0.0348 sec	9.21 KB
Testcase 2	Easy	Hidden case	Success	15	0.0369 sec	9.14 KB
Testcase 3	Easy	Sample case	Success	10	0.0501 sec	9.04 KB

No Comments

QUESTION 10

Correct Answer

Score 60

Check Types › Coding

Lists CS101

QUESTION DESCRIPTION

It is often required that programs need to be checked and guarded against invalid inputs.

Write a function 'check_types' that takes as parameter a list 'lst' and returns a list of all the data types that were present in the list that was passed as a parameter. Your function should also include guards against invalid arguments.

```
>>> print(check_types([]))
[]
>>> print(check_types(["Programming", "List", "Fundamentals"]))
['str']
>>> print(check_types(['hello', [2, [False, [3.5]]], 'world']))
['str', 'list', 'int', 'bool', 'float']
>>> print(check_types('this is not right'))
Error: Bad argument. Function 'check_types' only accept lists
```

INTERVIEWER GUIDELINES

```
def check_types(lst):
    li = []
    if type(lst) != list:
        return("Error: Bad argument. Function 'check_types' only accept lists")
    else:
        for i in lst:
            if type(i) == int:
                if "int" not in li:
                    li.append("int")
            elif type(i) == str:
                if "str" not in li:
```

```
            li.append("str")
        elif type(i) == list:
            if "list" not in li:
                li.append("list")
            li = recursive(i, li)
        elif type(i) == bool:
            if "bool" not in li:
                li.append("bool")
    return(li)
def recursive(s, li):
    for i in s:
        if type(i) == int:
            if "int" not in li:
                li.append("int")
        elif type(i) == str:
            if "str" not in li:
                li.append("str")
        elif type(i) == list:
            if "list" not in li:
                li.append("list")
            li = recursive(i, li)
        elif type(i) == bool:
            if "bool" not in li:
                li.append("bool")
        elif type(i) == float:
            if "float" not in li:
                li.append("float")
    return(li)
```

CANDIDATE ANSWER

Language used: **Python 3**

```
import ast
lst = input()
lst = ast.literal_eval(lst)

def check_types(lst):
    x = []
    if type (lst) != list and "list" not in x:
        return "Error: Bad argument. Function 'check_types' only accept
lists"
    else:
        for i in lst:
            if type (i) == str and "str" not in x:
                x.append("str")
            elif type (i) == bool and "bool" not in x:
                x.append("bool")
            elif type (i) == int and "int" not in x:
                x.append("int")
            elif type (i) == float and "float" not in x:
                x.append("float")
            elif type (i) == list:
                if "list" not in x:
                    x.append("list")
                y = check_types(i)
                for j in y:
                    if j not in x:
                        x.append(j)
        return (x)
```

```
print(check_types(lst))
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	Success	10	0.056 sec	9.05 KB
Testcase 1	Easy	Hidden case	Success	10	0.0541 sec	8.99 KB
Testcase 2	Easy	Sample case	Success	10	0.0383 sec	9.11 KB
Testcase 3	Easy	Hidden case	Success	10	0.037 sec	9.05 KB
Testcase 4	Easy	Sample case	Success	10	0.0534 sec	9.04 KB
Testcase 5	Easy	Sample case	Success	10	0.0546 sec	9.05 KB

No Comments

QUESTION 11

Correct Answer

Score 20

Problem Solving - Pattern 4 › Coding

QUESTION DESCRIPTION

Problem

Write an *iterative* function named `pattern` to generate the following pattern for a given parameter, `n`.

Sample

```
>>> pattern(3)
1
2 1
4 2 1

>>> pattern(1)
1

>>> pattern(2)
1
2 1

>>> pattern(6)
1
2 1
4 2 1
8 4 2 1
16 8 4 2 1
32 16 8 4 2 1

>>> pattern(8)
1
2 1
4 2 1
8 4 2 1
16 8 4 2 1
32 16 8 4 2 1
64 32 16 8 4 2 1
```

Input

Input `n` from the console without any prompt.

Constraints

- `isinstance(n, int)` is `True`

- n >= 1

CANDIDATE ANSWER

Language used: **Python 3**

```
n = int(input())
def pattern(n):
    s = ""
    for i in range (n):
        s = (str(2 ** i)) + " " + s
        print (s)

```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
TestCase 0	Easy	Sample case	Success	2.5	0.0365 sec	7.91 KB
TestCase 1	Easy	Hidden case	Success	2.5	0.0271 sec	8.13 KB
TestCase 2	Easy	Hidden case	Success	2.5	0.0244 sec	8.02 KB
TestCase 3	Easy	Sample case	Success	2.5	0.0289 sec	8.13 KB
Testcase 4	Easy	Sample case	Success	10	0.0202 sec	8 KB

No Comments

QUESTION 12

Correct Answer

Score 20

Problem Solving - Pattern 5 › Coding

QUESTION DESCRIPTION

Problem

Write an *iterative* function named `pattern` to generate the following pattern for a given parameter, `n`.

Sample

```
>>> pattern(3)
1
1 2 1
1 2 4 2 1

>>> pattern(1)
1

>>> pattern(2)
1
1 2 1

>>> pattern(6)
1
1 2 1
1 2 4 2 1
1 2 4 8 4 2 1
1 2 4 8 16 8 4 2 1
1 2 4 8 16 32 16 8 4 2 1

>>> pattern(8)
1
1 2 1
1 2 4 2 1
1 2 4 8 4 2 1
```

```
1 2 4 8 16 8 4 2 1
1 2 4 8 16 32 16 8 4 2 1
1 2 4 8 16 32 64 32 16 8 4 2 1
1 2 4 8 16 32 64 128 64 32 16 8 4 2 1
```

Input

Input `n` from the console without any prompt.

Constraints

- `isinstance(n, int)` is `True`
- n >= 1

CANDIDATE ANSWER

Language used: **Python 3**

```python
n = int(input())
def pattern(n):
    s = ""
    j = ""
    for i in range(n):
        s += str((2 ** (i))) + " "
        if i > 0:
            j = str((2 ** (i - 1))) + " " + j
        print (s + j)
pattern(n)
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
TestCase 0	Easy	Sample case	Success	2.5	0.0345 sec	7.97 KB
TestCase 1	Easy	Hidden case	Success	2.5	0.0409 sec	8.08 KB
TestCase 2	Easy	Hidden case	Success	2.5	0.0326 sec	8.16 KB
TestCase 3	Easy	Sample case	Success	2.5	0.0223 sec	8.04 KB
Testcase 4	Easy	Sample case	Success	10	0.0495 sec	8.2 KB

No Comments