1. Create a function that takes the width, height and character and returns a picture frame as a 2D list.

**Examples**

get\_frame(4, 5, "#") ➞ [

["####"],

["# #"],

["# #"],

["# #"],

["####"]

]

# Frame is 4 characters wide and 5 characters tall.

get\_frame(10, 3, "\*") ➞ [

["\*\*\*\*\*\*\*\*\*\*"],

["\* \*"],

["\*\*\*\*\*\*\*\*\*\*"]

]

# Frame is 10 characters and wide and 3 characters tall.

get\_frame(2, 5, "0") ➞ "invalid"

# Frame's width is not more than 2.

**Ans:**

def get\_frame(w, h, c):

if w==2:

print("invalid")

return

l=[]

for i in range(h):

string=""

l1=[]

for j in range(w):

if i==0 or i ==h-1 or j==0 or j==w-1:

string+=c

else:

string+=" "

l1.append(string)

l.append(l1)

print(l)

2. Write three functions:

1. boolean\_and

2. boolean\_or

3. boolean\_xor

These functions should evaluate a list of True and False values, starting from the leftmost element and evaluating pairwise.

**Examples**

boolean\_and([True, True, False, True]) ➞ False

# [True, True, False, True] => [True, False, True] => [False, True] => False

boolean\_or([True, True, False, False]) ➞ True

# [True, True, False, True] => [True, False, False] => [True, False] => True

boolean\_xor([True, True, False, False]) ➞ False

# [True, True, False, False] => [False, False, False] => [False, False] => False

**Ans:**

def boolean\_and(l):

result=False

for i in range(len(l)-1):

if i==0:

result=l[i] and l[i+1]

else:

result=result and l[i+1]

print(result)

def boolean\_or(l):

result=False

for i in range(len(l)-1):

if i==0:

result=l[i] or l[i+1]

else:

result=result or l[i+1]

print(result)

def boolean\_xor(l):

result=False

for i in range(len(l)-1):

if i==0:

result=(l[i] and not l[i+1]) or (not l[i] and l[i+1])

else:

result=(result and not l[i+1]) or (not result and l[i+1])

print(result)

3. Create a function that creates a box based on dimension n.

**Examples**

make\_box(5) ➞ [

"#####",

"# #",

"# #",

"# #",

"#####"

]

make\_box(3) ➞ [

"###",

"# #",

"###"

]

make\_box(2) ➞ [

"##",

"##"

]

make\_box(1) ➞ [

"#"

]

**Ans:**

def make\_box(n):

l=[]

for i in range(n):

string=""

l1=[]

for j in range(n):

if i==0 or i ==n-1 or j==0 or j==n-1:

string+="#"

else:

string+=" "

l1.append(string)

l.append(l1)

print(l)

4. Given a common phrase, return False if any individual word in the phrase contains duplicate letters. Return True otherwise.

**Examples**

no\_duplicate\_letters("Fortune favours the bold.") ➞ True

no\_duplicate\_letters("You can lead a horse to water, but you can't make him drink.") ➞ True

no\_duplicate\_letters("Look before you leap.") ➞ False

# Duplicate letters in "Look" and "before".

no\_duplicate\_letters("An apple a day keeps the doctor away.") ➞ False

# Duplicate letters in "apple", "keeps", "doctor", and "away".

**Ans:**

def no\_duplicate\_letters(string):

l=string.split(" ")

result=True

for i in l:

for j in i:

if i.count(j)>1:

result=False

break

print(result)

5. Write a regular expression that will match the states that voted yes to President Trump's impeachment. You must use RegEx positive lookahead.

Example

txt = "Texas = no, California = yes, Florida = yes, Michigan = no"

pattern = "yourregularexpressionhere"

re.findall(pattern, txt) ➞ ["California", "Florida"]

**Ans:**

txt = "Texas = no, California = yes, Florida = yes, Michigan = no"

pattern = r'([\w\.-]+) = yes'

re.findall(pattern, txt)