**Python Advance Programming Assignment-10**

**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.

In [64]:

**def** print\_rectangle(n, m,char) :

**for** i **in** range(1, n**+**1) :

**for** j **in** range(1, m**+**1) :

**if** (i **==** 1 **or** i **==** n **or**

j **==** 1 **or** j **==** m) :

print(char, end**=**"")

**else** :

print("", end**=**"")

print()

print\_rectangle(4, 5,'#')

#####

##

##

#####

**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**

In [12]:

**def** boolean\_and(l):

temp**=False**

**for** i **in** l:

**if** i**==**i**+**1:

temp**=True**

**else**:

temp**=False**

**return** temp

In [15]:

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

Out[15]:

False

**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) ➞ [

"#" ]

In [39]:

**def** makebox(rows):

**for** i **in** range(1, rows **+** 1):

**if** (i **==** 1 **or** i **==** rows):

**for** j **in** range(1, rows **+** 1):

print("\*", end **=** "")

**else**:

**for** j **in** range(1, rows **+** 1):

**if** (j **==** 1 **or** j **==** rows):

print("\*", end **=** "")

**else**:

print(end **=** " ")

print()

In [40]:

makebox(5)

\*\*\*\*\*

\* \*

\* \*

\* \*

\*\*\*\*\*

In [41]:

makebox(3)

\*\*\*

\* \*

\*\*\*

In [42]:

makebox(2)

\*\*

\*\*

In [43]:

makebox(1)

\*

**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".**

In [52]:

**def** no\_duplicate\_letters(str):

**for** i **in** range(len(str)):

**for** j **in** range(i **+** 1,len(str)):

**if**(str[i] **==** str[j]):

**return** **False**

**return** **True**

In [53]:

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

Out[53]:

False

**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"]