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:

In [6]:

```python
def get_frame(width,height,char):
    if width <= 2:
        print(f'get_frame({width},{height},{char}) ➞ invalid')
    else:
        output = []
        for h in range(height):
            if h == 0 or h == height-1:
                output.append([width*char])
            else:
                output.append([char+' '*(width-2)+char])
        print(f'get_frame({width},{height},{char}) ➞ [')
        for i in output:
            print(i)
        print(']')
get_frame(4, 5, "#")
get_frame(10, 3, "*")
get_frame(2, 5, "0")
```

```
get_frame(4,5,#) ➞ [
['####']
['#  #']
['#  #']
['#  #']
['####']
]
get_frame(10,3,*) ➞ [
['**********']
['*        *']
['**********']
]
get_frame(2,5,0) ➞ invalid
```

2. Write three functions:

boolean_and
boolean_or
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:

In [40]:

```
def boolean_and(list1):
    copy1 = list1
    for exp1 in range(len(list1)):
        new1 = list1[0] and list1[1]
        list1 = list1[2:len(list1)]
        list1.insert(0,new1)
    print(f'boolean_and({copy1}) ➞ {list1[0]}')
def boolean_or(list2):
    copy2 = list2
    for exp2 in range(len(list2)):
        new2 = list2[0] or list2[1]
        list2= list2[2:len(list2)]
        list2.insert(0,new2)
    print(f'boolean_and({copy2}) ➞ {list2[0]}')
def boolean_xor(list3):
    copy3 = list3
    while len(list3)!= 1:
        new3 = (list3[0])^(list3[1])
        list3= list3[2:len(list3)]
        list3.insert(0,new3)
    print(f'boolean_and({copy3}) ➞ {list3[0]}')
boolean_and([True, True, False, True])
boolean_or([True, True, False, False])
boolean_xor([True, True, False, False])
```

```
boolean_and([True, True, False, True]) ➞ False
boolean_and([True, True, False, False]) ➞ True
boolean_and([True, True, False, False]) ➞ 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) → [
"#"
]

Ans:

In [4]:

```
def make_box(n):
    output = []
    for i in range(n):
        if i == 0 or i == n-1:
            output.append(['#'*n])
        else:
            output.append(['#'+' '*(n-2)+'#'])
    print(f'make_box(n) → [')
    for i in output:
        print(i)
    print()
    print(']')
make_box(5)
make_box(3)
make_box(2)
make_box(1)
```

```
make_box(n) → [
['#####']
['#   #']
['#   #']
['#   #']
['#####']

]
make_box(n) → [
['###']
['# #']
['###']

]
make_box(n) → [
['##']
['##']

]
make_box(n) → [
['#']

]
```

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:

In [8]:
```
def no_duplicate_letters(phrase):
    output = None
    for word in phrase.split(' '):
        if len(word) == len(set(word)):
            output = True
        else:
            output = False
            break
    print(f'no_duplicate_letters({phrase}) ⟶ {output}')
no_duplicate_letters("Fortune favours the bold.")
no_duplicate_letters("You can lead a horse to water, but you can't make him drink.")
no_duplicate_letters("Look before you leap.")
no_duplicate_letters("An apple a day keeps the doctor away.")
```

```
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
no_duplicate_letters(An apple a day keeps the doctor away.) ⟶ 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.

Examples:
txt = "Texas = no, California = yes, Florida = yes, Michigan = no"
pattern = "yourregularexpressionhere"
re.findall(pattern, txt) ⟶ ["California", "Florida"]

Ans:

In [9]:
```
import re
txt = "Texas = no, California = yes, Florida = yes, Michigan = no"
pattern = r'\w+(?=\s=\syes*)'
match = re.findall(pattern,txt)
print(f're.findall{pattern,txt} ⟶ {match}')
```

```
re.findall('\\w+(?=\\s=\\syes*)', 'Texas = no, California = yes, Florida = yes, Michigan = no') ⟶ ['California', 'Florida']
```