Code Tracing (Variables) (10 points)

For each snippet of code below, write the value of each variable and its type.

	a	b	c
`a = "Nihar"`	"Nihar"/str		
`b = "Peter"`	"Nihar"/str	"Peter"/str	
`c = 2`	"Nihar"/str	"Peter"/str	2/int
`a = a * c + str(c)`	"NiharNihar2"/str	"Peter"/str	2/int
`b = b + str(c - 1)`	"NiharNihar2"/str	"Peter1"/str	2/int

	n	judge	r
`# user enters the number "7.2"` `n = input("Temperature: ")`	"7.2"/str		
`n = float(n) // 1`	7.0/float (-0 for int)		
`judge = n < 40`	7.0/float	True/boolean	
`judge = not judge or n > 7`	7.0/float	False/boolean	
`r = "Temp is: " + str(judge)`	7.0/float	False/boolean	"Temp is: False"/str

Code Tracing (Output) (12 points)

Write the output of the code snippets below:

```
s = 4
e = 10
times = 0
for num in range(s, e):
    if num + s < e:
        print("pow!" + str(num))
    times = times + 1
print(times)
```

Output

pow!4
pow!5
6

```
w = "purple"
x = 0
for index in range(len(w)):
    print(w[x])
    print(w[index])
```

Output

p
p
p
u
p
r
p
p
p
l
p
e

`p1 = "hermione"` `p2 = "hogwarts"` `if len(p1) == len(p2) and p1[0] == p2[0]:` `    print("Here's your letter")` `    p1 = "harry"` `if p1[0] == p2[0] and len(p1) < len(p2):` `    print("Hoot! Owl! Hoot!")`	Output *Here's your letter* *Hoot! Owl! Hoot!*

Function Mysteries (14 points)

All functions must either return or print something.
You may not use any casting functions.
You may <u>only</u> use the following options as values for the parameters, <u>each one a maximum of one time</u>. You will not use all values in this table.

True **False**	**2** **5** **8** **11**	**"narwhal"** **"art"** **"dandelion"**

`def func3(m1, m2, a):` `    if m1 and m2:` `        return a * 1.0` `    elif m2 and a < 10:` `        return a / 2` `    else:` `        return (a + 1) / 2`	*# one example* *r = func3(True, False, 2)* *# another example* *r = func3(False, True, 8)*	*1.5*	
`def func4(x, y):` `    for i in range(x, y):` `        j = (y - x) - i` `        print(str(i) + ":" + str(j))`	*# one example* *func4(2, 5)*		*2:1* *3:0* *4:-1*
`def func5(b, c):` `    s = b` `    t = c` `    if len(c) > len(b):` `        s = c` `        t = b` `    r = ""` `    for i in range(len(s) - len(t)):` `        r = r + s[i + len(t) - 1]` `    return r`	*# one example* *func5("art", "narwhal")*	*"rwha"*	

Writing Functions (22 points)

1) Write a function, `name_box`, that takes one parameter, `name`, and prints out a name box of the specified size. Notice that when the name has an even length,there is no middle row so the name appears on the row “after” the middle.

Examples:

Function call	output	Function call	output
`name_box("")`	`**` `**`	`name_box("K")`	`***` `*K*` `***`
`name_box("An")`	`****` `*  *` `*An*` `****`	`name_box("Ace")`	`*****` `*   *` `*Ace*` `*   *` `*****`
`name_box("Fran")`	`******` `*    *` `*    *` `*Fran*` `*    *` `******`	`name_box("Frank")`	`*******` `*     *` `*     *` `*Frank*` `*     *` `*     *` `*******`

```
def name_box(name):
    top_bottom = "*" + ("*" * len(name)) + "*"
    print(top_bottom)
    for row_num in range(len(name)):
        if row_num == len(name) // 2:
            print("*" + name + "*")
        else:
            print("*" + (" " * len(name)) + "*")
    print(top_bottom)
```

(24 points)

2) Write a function, `rotate`, that takes a string, `target`, and an integer, `n`, and rotates the string the desired amount before **returning** it. A rotation is performed by taking the `n` characters at the beginning of the string and placing them, in the same order, at the end of the string.

You may not use string slicing to implement this function. You may not use any string methods other than the `[]` accessor and `+` to concatenate strings.
Examples:

Function call (underlining used to emphasize how the letters change)	Return value (you do not need to reproduce underlines)
`rotate("hello", 1)`	`"elloh"`
`rotate("hello", 3)`	`"lohel"`
`rotate("Cat curling", 8)`	`"ingCat curl"`
`rotate("Peter Rock", 6)`	`"RockPeter "`
`rotate("hello", 0)`	`"hello"`

Notice that a rotation of 0 does not change the string. You may assume that `0 <= n < len(target)`.

```
def rotate(target, n):
    f = ""
    for i in range(n):
        f += target[i]
    s = ""
    for j in range(n, len(target)):
        s += target[j]
    return s + f
```

Writing a Program (18 points)

Write a complete program, including a `main` function and a call to `main` after you have defined it, that asks the user how many labels they would like to make, then, for each label, asks the user 1) what name to print on the label and 2) the rotation before printing the rotated name in a box of stars to create the label.

You may find calling one or more of your previous functions useful (you don't need to copy them down here again). You may assume you have access to correct implementations of all functions in the "Writing Functions" section.

Example 1: output (user input **underlined bold**)	Example 2: output
How many labels? **2** What name? **Chai** Rotation? **1** ****** * * * * *haiC* * * ****** What name? **Ben** Rotation? **0** ***** * * *Ben* * * *****	How many labels? **3** What name? **Cat** Rotation? **2** ***** * * *tCa* * * ***** What name? **T** Rotation? **0** *** *T* *** What name? **C** Rotation? **0** *** *C* ***

```
def main():
    num = int(input("How many labels? "))
    for i in range(num):
        name = input("What name? ")
        rot = int(input("Rotation? "))
        rot_name = rotate(name, rot)
        name_box(rot_name)

if __name__ == "__main__":
    main()
```