In [1]:
```
'''QUESTION ONE
Write a program to produce the following output using for loop
+----+
\ /
/ \
\ /
/ \
\ /
/ \
+----+'''
```

Out[1]:
```
'QUESTION ONE\nWrite a program to produce the following output using for loop
\n+----+\n\\ /\n/ \\ /\n/ \\ /\n/ +----+'
```

In [2]:
```
for i in range(5):
    if i % 6 == 0:
        print("+----+")
    else:
        print("\\    /")
        print("/    \\")
print("+----+")

# Output:
# +----+
# \    /
# /    \
# \    /
# /    \
# +----+
```

```
+----+
\    /
/    \
\    /
/    \
\    /
/    \
\    /
/    \
+----+
```

In [3]:
```
'''.

QUETION TWO
Write a program to produce the following output using for loop
**********
**********
**********
**********
**********'''
```

Out[3]:
```
'. \n\nQUETION TWO\nWrite a program to produce the following output using for
loop\n**********\n**********\n**********\n**********\n**********'
```

In [4]:
```
for i in range(5):
    print("**********")
```

```
**********
**********
**********
**********
**********
```

In [5]:
```
'''
QUETION THREE
Complete the code for the following for loop:
for in range(1,7):'''
```

Out[5]:
```
'\nQUETION THREE\nComplete the code for the following for loop:\nfor in range
(1,7):'
```

In [6]:
```
for i in range(1, 7):
    a = i
    b = i * 2
    c = i * 7 - 3
    d = i * 10 - 10
    e = i * 2 - 9
    f = i * 15 + 77
    g = i * -3 - 1

    print(f"{a} {b} {c} {d} {e} {f} {g}")
```

```
1 2 4 0 -7 92 -4
2 4 11 10 -5 107 -7
3 6 18 20 -3 122 -10
4 8 25 30 -1 137 -13
5 10 32 40 1 152 -16
6 12 39 50 3 167 -19
```

In [7]:
```
'''.
QUESTION FOUR
Write a program to produce the following output using for loops. Then
use a class constant to make it possible to change the number of lines in
the figure.
1
22
333
4444
55555
666666
7777777'''
```

Out[7]:
```
'. \nQUESTION FOUR\nWrite a program to produce the following output using for
loops. Then\nuse a class constant to make it possible to change the number of
lines in\nthe figure.\n1\n22\n333\n4444\n55555\n666666\n7777777'
```

In [8]:
```
def print_figure(num_lines):
    for i in range(1, num_lines + 1):
        line = str(i) * i
        print(line)

# Example usage:
num_lines = 7
print_figure(num_lines)
```

```
1
22
333
4444
55555
666666
7777777
```

In [9]:
```
'''
QUESTION FIVE
Write a method named pay that accepts two parameters: a real number
for a TA's salary, and an integer for the number of hours the TA worked
this week. The method should return how much money to pay the TA.
For example, the call
pay(5.50, 6)
should return
33.0.
The TA should receive "overtime" pay of 1 ½ normal salary for any hours
above 8. For example, the call pay(4.00, 11) should return (4.00 *
8) + (6.00 * 3) or 50.0.

'''
```

Out[9]:
```
'\nQUESTION FIVE\nWrite a method named pay that accepts two parameters: a rea
l number\nfor a TA\'s salary, and an integer for the number of hours the TA w
orked\nthis week. The method should return how much money to pay the TA.\nFor
example, the call\npay(5.50, 6)\nshould return\n33.0.\nThe TA should receive
"overtime" pay of 1 ½ normal salary for any hours\nabove 8. For example, the
call pay(4.00, 11) should return (4.00 *\n8) + (6.00 * 3) or 50.0. \n\n'
```

In [10]:
```
def pay(hourly_rate, hours_worked):
    regular_hours = min(hours_worked, 8)
    overtime_hours = max(0, hours_worked - 8)

    regular_pay = regular_hours * hourly_rate
    overtime_pay = overtime_hours * (hourly_rate * 1.5)

    total_pay = regular_pay + overtime_pay
    return total_pay

# Example usage:
hourly_rate1 = 5.50
hours_worked1 = 6
print(pay(hourly_rate1, hours_worked1))  # Output: 33.0

hourly_rate2 = 4.00
hours_worked2 = 11
print(pay(hourly_rate2, hours_worked2))  # Output: 50.0
```

```
33.0
50.0
```

In [11]:
```
'''
QUESTION SIX

Consider the following method for converting milliseconds into days:
// converts milliseconds to days
def toDays(millis):
 return millis / 1000.0 / 60.0 / 60.0 / 24.0
Write a similar method named area that takes as a parameter the radius of
a circle and that returns the area of the circle. For example, the call
area(2.0);
should return
12.566370614359172.
Recall that area can be computed as π times the radius squared and that
Python has a constant called math.pi'''
```

Out[11]:
```
'\nQUESTION SIX\n\nConsider the following method for converting milliseconds
into days:\n// converts milliseconds to days\ndef toDays(millis):\n return mi
llis / 1000.0 / 60.0 / 60.0 / 24.0\nWrite a similar method named area that ta
kes as a parameter the radius of\na circle and that returns the area of the c
ircle. For example, the call\narea(2.0);\nshould return\n12.56637061435917
2.\nRecall that area can be computed as π times the radius squared and that\n
Python has a constant called math.pi'
```

In [12]:
```
import math

def area(radius):
    return math.pi * radius ** 2

# Example usage:
radius = 2.0
result = area(radius)
print(result)  # Output: 12.566370614359172
```

```
12.566370614359172
```

In [13]:
```
'''
QUESTION SIX

Copy and paste the following code into pythons IDLE script environment.
low = 1
high = 1001
sum = 0
for i in range(low,high):
sum += i
print("sum = " , sum)
Modify the code to use a input to prompt the user for the values of low and hig
low? 1
high? 1001
sum = 500500
Below is an execution with different values for low and high:
low? 300
high? 5298
sum = 13986903
You should exactly reproduce this format.

'''
```

Out[13]:
```
'\nQUESTION SIX\n\nCopy and paste the following code into pythons IDLE script
environment.\nlow = 1\nhigh = 1001\nsum = 0\nfor i in range(low,high):\nsum +
= i\nprint("sum = " , sum)\nModify the code to use a input to prompt the user
for the values of low and high. Below is a sample execution in which the user
asks for the same values as in the original program (1 through 1000):\nlow? 1
\nhigh? 1001\nsum = 500500\nBelow is an execution with different values for l
ow and high:\nlow? 300\nhigh? 5298\nsum = 13986903\nYou should exactly reprod
uce this format.\n\n'
```

In [14]:
```
low = int(input("low? "))
high = int(input("high? "))
sum_result = 0

for i in range(low, high):
    sum_result += i

print("sum =", sum_result)
```

```
low? 1
high? 8
sum = 28
```

In [15]:
```
'''Write a program using while loop that prompts the user for numbers until the
```

Out[15]:
```
'Write a program using while loop that prompts the user for numbers until the
user types 0, then outputs their sum.'
```

In [16]:
```
sum_result = 0

while True:
    num = int(input('Please enter a number or 0 to finish: '))

    if num == 0:
        break

    sum_result += num

print('The total number entered is =', sum_result)
```

```
Please enter a numberor 0 to finish: 2
Please enter a numberor 0 to finish: 8
Please enter a numberor 0 to finish: 9
Please enter a numberor 0 to finish: 0
The total number entered is = 19
```

In [17]:
```
#Write a program using while loop that prompts the user for numbers until the u
```

In [20]:
```
sum = 0

while True:
    num =int(input('Please enter a number or  -1 to terminate: '))

    if num ==-1:
        break

    sum = sum + num

print('The total number entered is = ', sum)
```

```
Please enter a number or-1 to terminate: 2
Please enter a number or-1 to terminate: 4
Please enter a number or-1 to terminate: -1
The total number entered is =  6
```

In [21]:
```
'''
    QUESTION TEN
Write a method named repl that accepts a String and a number of repetitions as
and returns the String concatenated that many times. For example, the call repl
returns "hellohellohello". If the number of repetitions is 0 or less, an empty
```

Out[21]:
```
'Write a method named repl that accepts a String and a number of repetitions
as parameters \nand returns the String concatenated that many times. For exam
ple, the call repl("hello", 3) \nreturns "hellohellohello". If the number of
repetitions is 0 or less, an empty string is returned.'
```

In [25]:
```
def repl(input_str, repetitions):
    if repetitions <= 0:
        return ""

    result = input_str * repetitions
    return result

# Example usage:
string_input = input('Please enter a word:  ')
repetition_count = int(input('enter the number of times you want to repeat the
result = repl(string_input, repetition_count)
print(result)
```

```
Please enter a word:  joy
enter the number of times you want to repeat the word: 8
joyjoyjoyjoyjoyjoyjoyjoy
```

In [ ]:
```
'''
QUESTION ELEVEN

Write a method called printRange that accepts two integers as arguments and pri
printRange(2, 7)
printRange(19, 11)
printRange(5, 5)
The output produced should be the following:
2 3 4 5 6 7
19 18 17 16 15 14 13 12 11
5'''
```

In [28]:
```
def printRange(start, end):
    if start < end:
        # Increasing sequence
        for num in range(start, end + 1):
            print(num, end=" ")
    elif start > end:
        # Decreasing sequence
        for num in range(start, end - 1, -1):
            print(num, end=" ")
    else:
        # Numbers are the same
        print(start)

# Sample calls
printRange(1, 10)
```

```
1 2 3 4 5 6 7 8 9 10
```

In [ ]:
```
'''
QUESTION TWELVE

Write a method named smallestLargest that asks the user to enter numbers, then
How many numbers do you want to enter? 4
Number 1: 5
Number 2: 11
Number 3: -2
Number 4: 3
Smallest = -2
Largest = 11'''
```

In [29]:
```
def smallestLargest():
    # Get the number of numbers to read from the user
    num_count = int(input("How many numbers do you want to enter? "))

    # Ensure the user enters a valid number greater than 0
    while num_count <= 0:
        print("Please enter a valid number greater than 0.")
        num_count = int(input("How many numbers do you want to enter? "))

    # Initialize variables to keep track of the smallest and largest numbers
    smallest = float('inf')  # Initialize to positive infinity
    largest = float('-inf')  # Initialize to negative infinity

    # Loop to get input numbers from the user
    for i in range(1, num_count + 1):
        number = float(input(f"Number {i}: "))
        smallest = min(smallest, number)
        largest = max(largest, number)

    # Print the smallest and largest numbers
    print(f"Smallest = {smallest}")
    print(f"Largest = {largest}")

# Call the method
smallestLargest()
```

```
How many numbers do you want to enter? 5
Number 1: 1
Number 2: 3
Number 3: 5
Number 4: 6
Number 5: 9
Smallest = 1.0
Largest = 9.0
```

In [ ]:
```
'''
QUESTION THIRTEEN
Write a method called printAverage that uses a sentinel loop to repeatedly prom
Type a number: 7
Type a number: 4
Type a number: 16
Type a number: -4
Average was 9.0
If the first number that the user types is negative, do not print an average:
Type a number: -2'''
```

In [30]:
```
def printAverage():
    total = 0
    count = 0

    while True:
        number = float(input("Type a number: "))

        if number < 0:
            break  # Exit the loop if a negative number is entered

        total += number
        count += 1

    if count > 0:
        average = total / count
        print(f"Average was {average}")
    else:
        print("No nonnegative numbers were entered.")

# Call the method
printAverage()
```

```
Type a number: 7
Type a number: 0
Type a number: 89
Type a number: 67
Type a number: 56
Type a number: 78
Type a number: 6
Type a number: 3
Type a number: 3
Type a number: 3
Type a number: 4
Type a number: -1
Average was 28.727272727272727
```

In [ ]:
```
'''
QUETION FOURTEEN

Write a method named numUnique that takes three integers as parameters and
returns the number of unique integers among the three. For example, the call nu
should return 3 because the parameters have three different values. By contrast
should return 2 because there are only two unique numbers among the three param
```

In [33]:
```
def numUnique(num1, num2, num3):
    unique_numbers = set([num1, num2, num3])
    return len(unique_numbers)

# Examples
result1 = numUnique(18, 3, 4)
result2 = numUnique(6, 7, 6)

print(result1)  # Output: 3
print(result2)  # Output: 2
```

```
3
2
```

In [ ]:
```
'''
QUESTION FIFTEEN
A Random object generates pseudo-random numbers. Find out how to use the Random
2 + 4 = 6
3 + 5 = 8
5 + 6 = 11
1 + 1 = 2
4 + 3 = 7
You won after 5 tries!'''
```

In [34]:
```python
import random

def roll_dice():
    return random.randint(1, 6)

def simulate_dice_rolls():
    target_sum = 7
    tries = 0

    while True:
        dice1 = roll_dice()
        dice2 = roll_dice()

        total = dice1 + dice2
        print(f"{dice1} + {dice2} = {total}")

        tries += 1

        if total == target_sum:
            print(f"You won after {tries} tries!")
            break

# Call the method to simulate dice rolls
simulate_dice_rolls()
```

```
3 + 6 = 9
5 + 1 = 6
3 + 6 = 9
3 + 2 = 5
4 + 5 = 9
6 + 5 = 11
5 + 2 = 7
You won after 7 tries!
```

In [ ]:

In [ ]:

In [ ]: