In [1]:
```
2//3
```
Out[1]:
```
0
```

In [2]:
```
6<<2
```
Out[2]:
```
24
```

In [3]:
```
6&2
```
Out[3]:
```
2
```

In [4]:
```
6|2
```
Out[4]:
```
6
```

In [ ]:
```
#What does the finally keyword denotes in python
#It encloses the lines of code which will be executed if any error occurs while
#the try block.
```

In [ ]:
```
#What does raise keyword is used for in python?
#It is used to raise an exception
```

In [ ]:
```
#common use case of yield keyword in python?
#in defining a generator
```

In [ ]:
```
#Choose all the correct options to answer your question.
#Which of the following are the valid variable names?
#A) _abc B) 1abc
#C) abc2 D) None of the above Ans. None of the above
#Which of the following are the keywords in python?
#A) yield B) raise
#C) look-in D) all of the aboven   Ans.All of the above
```

In [*]:

```
#Write a python program to find the factorial of a number?
def factorial(n):
    if n == 0:
        return 1
    else:
        return n * factorial(n - 1)

# Input number whose factorial is to be calculated
number = int(input("Enter a number: "))


# Check if the number is negative
if number < 0:
    print("Factorial is not defined for negative numbers.")
elif number == 0:
    print("Factorial of 0 is 1")
else:
    result = factorial(number)
    print(f"Factorial of {number} is {result}")
```

In [*]:

```
# Write a program to find whether a number is prime or composite:

def is_prime(n):
    if n <= 1:
        return False  # 1 and numbers less than 1 are not prime numbers
    elif n <= 3:
        return True   # 2 and 3 are prime numbers
    elif n % 2 == 0 or n % 3 == 0:
        return False  # Numbers divisible by 2 or 3 are not prime numbers
    i = 5
    while i * i <= n:
        if n % i == 0 or n % (i + 2) == 0:
            return False  # Numbers divisible by any other number
```

In [ ]:

```
#Write a python program to check whether a given string is palindrome or not>
def is_palindrome(s):
    # Convert the string to lowercase and remove non-alphanumeric characters
    s = ''.join(e for e in s if e.isalnum()).lower()
    # Check if the string equals its reverse
    return s == s[::-1]

# Example usage:
input_string = input("Enter a string: ")
if is_palindrome(input_string):
    print(f"{input_string} is a palindrome.")
else:
    print(f"{input_string} is not a palindrome.")
```

In [ ]:

```
#Write a Python program to get the third side of right-angled triangle from two
def find_third_side(side1, side2):

    # Calculate the square of each side
    side1_squared = side1 ** 2
    side2_squared = side2 ** 2

    # Use Pythagorean theorem to find the length of the third side (hypotenuse)
    side3 = (side1_squared + side2_squared) ** 0.5

    return side3

# Example usage:
side1 = float(input("Enter length of first side: "))
side2 = float(input("Enter length of second side: "))

third_side = find_third_side(side1, side2)
print(f"The length of the third side (hypotenuse) is: {third_side}")
```

In [ ]:

```
#Write a python program to print the frequency of each of the characters preser
def count_character_frequency(input_string):
    # Initialize an empty dictionary to store character frequencies
    char_frequency = {}

    # Iterate through each character in the input string
    for char in input_string:
        # Increment the count of the character in the dictionary
        if char in char_frequency:
            char_frequency[char] += 1
        else:
            char_frequency[char] = 1

    return char_frequency

# Example usage:
input_string = input("Enter a string: ")

# Get the character frequency dictionary
frequency_dict = count_character_frequency(input_string)

# Print the character frequencies
print("Character frequencies:")
for char, count in frequency_dict.items():
    print(f"{char}: {count}")
```