In [10]:

```
# Question number: 1
# Write a function that accepts a positive integer as a parameter and then returns a
# representation of that number in binary (base 2).
# Hint: This is in many ways a trick question. Think!

# Function to convert a positive integer to its binary representation
def for_number (number):
    # Use the built-in bin() function to get the binary representation of the number
    # [2:] is used to remove the '0b' prefix from the binary string
    binary_number=bin(number)[2:]
    return binary_number

# Prompt the user to input a positive integer
number= int(input("Enter any postiver interger for binary representation: "))

# Call the function to convert the number to binary
binary_number= for_number(number)

# Call the function to convert the number to binary
print(f"The binary representation of {number} is: {binary_number}")
```

```
The binary representation of 45 is: 101101
```

In [23]:

```
# Question number: 2
# Write and test a function that takes an integer as its parameter and returns the
# factors of that integer. (A factor is an integer which can be multiplied by another to
# yield the original).

# Function that takes an integer as input and returns its factors
def interger_num (num1):
    # List to store the factors
    factors=[]

    # Loop through all numbers from 1 to num1
    for i in range(1, num1+1):
        # Check if num1 is divisible by i (i.e., if i is a factor)
        if num1 % i == 0:
            # Append the factor to the list
            factors.append(i)
    # Return the list of factors
    return factors

# prompt user input for the integer
num1=int(input("Enter any  interger:"))


# Call the function to get the factors of the input integer
factors= interger_num(num1)

# Print the factors of the input integer
print(f"The factor of {num1} is: {factors}")
```

```
The factor of 12 is: [1, 2, 3, 4, 6, 12]
```

In [2]:

```
# Question number: 3
# Write and test a function that determines if a given integer is a prime number. A
# prime number is an integer greater than 1 that cannot be produced by multiplying
# two other integers.

def for_number (number):

    if number<0:
        print(f"The given integer {number} is not a prime number.")
        return
    for i in range(2,number):
        if number%i==0:
            print(f"The given integer {number} is not a prime number.")
            return

    print(f"The given interger is a prime number.")

number=int(input("Enter any number of choice: "))
number=for_number(number)
```

```
The given interger is a prime number.
```

In [6]:

```
# Question number:4
# Computers are commonly used in encryption. A very simple form of encryption
# (more accurately "obfuscation") would be to remove the spaces from a message
# and reverse the resulting string. Write, and test, a function that takes a string
# containing a message and "encrypts" it in this way.
# Function to get the message from the user
def get_message():
    # Prompt the user to input a message
    message= input("Enter a message for encryption:")
    # Return the entered message
    return message

# Function to perform encryption by removing spaces and reversing the string
def for_encryption():
    # Get the message using the get_message function
    message=get_message()

    # Remove all spaces from the message
    for_spaces = message.replace(" ","")

    # Reverse the resulting string
    for_reverse= for_spaces[::-1]
    # Print the "encrypted" (obfuscated) message
    print(for_reverse)
    return

# Call the for_encryption function to execute the encryption process
for_encryption()
```

```
noitpyrcnenidesuylnommocerasretupmoC
```

In [12]:

```
# Question number: 5
# Another way to hide a message is to include the letters that make it up within
# seemingly random text. The letters of the message might be every fifth character,
# for example. Write and test a function that does such encryption. It should
# randomly generate an interval (between 2 and 20), space the message out
# accordingly, and should fill the gaps with random letters. The function should
# return the encrypted message and the interval used.
# For example, if the message is "send cheese", the random interval is 2, and for
# clarity the random letters are not random:
# send cheese
# s e n d c h e e s e
# sxyexynxydxy cxyhxyexyexysxye

# Function Prompt the user to enter a string and return the input
def inputstring():
    words=input("Enter the string: ")
    return words
# Function Add two spaces between each character of the input string
def space(word):
    word='  '.join(word)
    return word

# Function Add the string 'hjkyui' between each character of the input string
def encryption(word):
    word='hjkyui'.join(word)
    return word

# Call the inputstring function to get the user input and store it in 'word'
word = inputstring()

# Call the space function to add spaces between characters and print the result
print(space(word))

# Call the encryption function to add 'hjkyui' between characters and print the result
print(encryption(word))
```

```
p  r  i  n  t     c  o  m  p  u  t  e  r     s  c  i  e  n  c  e
phjkyuirhjkyuiihjkyuinhjkyuithjkyui hjkyuichjkyuiohjkyuimhjkyuiphjkyuiuhjkyuithjkyuiehjkyuirhjkyui hjkyuishjkyui
chjkyuiihjkyuiehjkyuinhjkyuichjkyuie
```

In [13]:

```
# Question number: 6
# Write a program that decrypts messages encoded as above

#Function Prompts the user to enter a string and return the input
def inputstring():
    words=input("Enter the string: ")
    return words

#Function Add three spaces between each character of the input string
def addspace(word):
    word='   '.join(word)
    return word

#FUnction  encrypt the string by inserting the sequence 'pano' between each character
def encryption(word):
    word='pano'.join(word)
    return word

#Functions remove double spaces and sequence"pano" from input string
def decrypt(word):
    removeword=word.replace('  ',"")
    decode=removeword.replace('pano',"")
    return decode

# Prompt the user for input and store it in the variable 'word'
word = inputstring()

# Decrypt the input string, remove 'pano' and double spaces, and print the result
print(decrypt(word))

# Add three spaces between characters of the input string and print the result
print(addspace(word))

# Encrypt the input string by adding 'pano' between characters and print the result
print(encryption(word))
```

```
computer science
c   o   m   p   u   t   e   r       s   c   i   e   n   c   e
cpanoopanompanoppanoupanotpanoepanorpano panospanocpanoipanoepanonpanocpanoe
```

In [ ]:

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js