The execution of a program can be controlled based on particular criteria using either chained conditionals or nested conditionals. The major distinction between these two methods is that a nested conditional requires nesting an if statement inside of another if statement, but a chained conditional involves a succession of if-else statements, each of which is connected to the next one in a chain.

def check\_number(number):

if number > 0:

return "positive"

elif number < 0:

return "negative"

else:

return "zero"

print(check\_number(10))

print(check\_number(-5))

print(check\_number(0))

The check\_number function in this example accepts a single argument, number, and produces a string based on that argument's value. The function determines the value of a number using a series of if-elif-else statements and returns the appropriate string.

def check\_age\_and\_citizenship(age, is\_citizen):

if age >= 18:

if is\_citizen:

return "Can vote"

else:

return "Cannot vote"

else:

return "Too young"

print(check\_age\_and\_citizenship(25, True))

print(check\_age\_and\_citizenship(17, True))

print(check\_age\_and\_citizenship(25, False))

In this illustration, the check age and citizenship function receives the two arguments age and is citizen as inputs, and it produces a string based on the results of those two arguments. The function checks a person's eligibility to vote using nested if-else expressions.

One method for avoiding nested conditionals is to split the conditionals into smaller, independent conditions that may be tested independently. For instance, you can utilise boolean variables to hold interim outcomes rather than having numerous nested if statements.

def check\_age\_and\_citizenship(age, is\_citizen):

eligible = False

if age >= 18:

eligible = True

if eligible and is\_citizen:

return "Can vote"

elif eligible:

return "Cannot vote"

else:

return "Too young"

print(check\_age\_and\_citizenship(25, True))

print(check\_age\_and\_citizenship(17, True))

print(check\_age\_and\_citizenship(25, False))

In this illustration, the function check age and citizenship makes use of a boolean variable eligible to hold the age check's intermediate result. This makes the code simpler to read and maintain because the function is no longer nested.

What purpose can variables serve in programming, and how may they be used to change values and store data?