Python has a set of keywords that are reserved words that cannot be used as variable names, function names, or any other identifiers:

|  |  |
| --- | --- |
| **Keyword** | **Description** |
| [and](https://www.w3schools.com/python/ref_keyword_and.asp) | A logical operator |
| [as](https://www.w3schools.com/python/ref_keyword_as.asp) | To create an alias |
| [assert](https://www.w3schools.com/python/ref_keyword_assert.asp) | For debugging |
| [break](https://www.w3schools.com/python/ref_keyword_break.asp) | To break out of a loop |
| [class](https://www.w3schools.com/python/ref_keyword_class.asp) | To define a class |
| [continue](https://www.w3schools.com/python/ref_keyword_continue.asp) | To continue to the next iteration of a loop |
| [def](https://www.w3schools.com/python/ref_keyword_def.asp) | To define a function |
| [del](https://www.w3schools.com/python/ref_keyword_del.asp) | To delete an object |
| [elif](https://www.w3schools.com/python/ref_keyword_elif.asp) | Used in conditional statements, same as else if |
| [else](https://www.w3schools.com/python/ref_keyword_else.asp) | Used in conditional statements |
| [except](https://www.w3schools.com/python/ref_keyword_except.asp) | Used with exceptions, what to do when an exception occurs |
| [False](https://www.w3schools.com/python/ref_keyword_false.asp) | Boolean value, result of comparison operations |
| [finally](https://www.w3schools.com/python/ref_keyword_finally.asp) | Used with exceptions, a block of code that will be executed no matter if there is an exception or not |
| [for](https://www.w3schools.com/python/ref_keyword_for.asp) | To create a for loop |
| [from](https://www.w3schools.com/python/ref_keyword_from.asp) | To import specific parts of a module |
| [global](https://www.w3schools.com/python/ref_keyword_global.asp) | To declare a global variable |
| [if](https://www.w3schools.com/python/ref_keyword_if.asp) | To make a conditional statement |
| [import](https://www.w3schools.com/python/ref_keyword_import.asp) | To import a module |
| [in](https://www.w3schools.com/python/ref_keyword_in.asp) | To check if a value is present in a list, tuple, etc. |
| [is](https://www.w3schools.com/python/ref_keyword_is.asp) | To test if two variables are equal |
| [lambda](https://www.w3schools.com/python/ref_keyword_lambda.asp) | To create an anonymous function |
| [None](https://www.w3schools.com/python/ref_keyword_none.asp) | Represents a null value |
| [nonlocal](https://www.w3schools.com/python/ref_keyword_nonlocal.asp) | To declare a non-local variable |
| [not](https://www.w3schools.com/python/ref_keyword_not.asp) | A logical operator |
| [or](https://www.w3schools.com/python/ref_keyword_or.asp) | A logical operator |
| [pass](https://www.w3schools.com/python/ref_keyword_pass.asp) | A null statement, a statement that will do nothing |
| [raise](https://www.w3schools.com/python/ref_keyword_raise.asp) | To raise an exception |
| [return](https://www.w3schools.com/python/ref_keyword_return.asp) | To exit a function and return a value |
| [True](https://www.w3schools.com/python/ref_keyword_true.asp) | Boolean value, result of comparison operations |
| [try](https://www.w3schools.com/python/ref_keyword_try.asp) | To make a try...except statement |
| [while](https://www.w3schools.com/python/ref_keyword_while.asp) | To create a while loop |
| with | Used to simplify exception handling |
| yield | To end a function, returns a generator |

**Python Keywords**

Keywords are the reserved words in Python.

We cannot use a keyword as a [variable](https://www.programiz.com/python-programming/variables-datatypes) name, [function](https://www.programiz.com/python-programming/function) name or any other identifier. They are used to define the syntax and structure of the Python language.

In Python, keywords are case sensitive.

There are 33 keywords in Python 3.7. This number can vary slightly over the course of time.

All the keywords except True, False and None are in lowercase and they must be written as they are. The list of all the keywords is given below.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| False | await | else | import | pass |
| None | break | except | in | raise |
| True | class | finally | is | return |
| and | continue | for | lambda | try |
| as | def | from | nonlocal | while |
| assert | del | global | not | with |
| async | elif | if | or | yield |

# This program prints Hello, world!

print('Hello, world!')

2. # This program adds two numbers

num1 = 1.5

num2 = 6.3

# Add two numbers

sum = float(num1) + float(num2)

# Display the sum

print('The sum of {0} and {1} is {2}'.format(num1, num2, sum))

Changing this operator, we can subtract **(-)**, multiply **(\*)**, divide **(/)**, floor divide **(//)** or find the remainder (%) of two numbers.

In this program, we asked the user to enter two numbers and this program displays the sum of two numbers entered by user.

We use the built-in function input() to take the input. input() returns a string, so we convert it into number using the float() function.

Alternative to this, we can perform this addition in a single statement without using any variables as follows.

print('The sum is %.1f' %(float(input('Enter first number: ')) + float(input('Enter second number: '))))

3. # Python Program to calculate the square root

# Note: change this value for a different result

num = 8

# To take the input from the user

#num = float(input('Enter a number: '))

num\_sqrt = num \*\* 0.5

print('The square root of %0.3f is %0.3f'%(num ,num\_sqrt))

# Find square root of real or complex numbers

# Importing the complex math module

import cmath

num = 1+2j

# To take input from the user

#num = eval(input('Enter a number: '))

num\_sqrt = cmath.sqrt(num)

print('The square root of {0} is {1:0.3f}+{2:0.3f}j'.format(num ,num\_sqrt.real,num\_sqrt.imag))

In this program, we use the sqrt() function in the cmath (complex math) module.

Notice that we have used the eval() function instead of float() to convert complex numbers as well. Also, notice the way in which the output is formatted.

# Python program to swap two variables

x = 5

y = 10

# To take inputs from the user

#x = input('Enter value of x: ')

#y = input('Enter value of y: ')

# create a temporary variable and swap the values

temp = x

x = y

y = temp

print('The value of x after swapping: {}'.format(x))

print('The value of y after swapping: {}'.format(y))

# Program to generate a random number between 0 and 9

# importing the random module

import random

print(random.randint(0,9))

# Taking kilometers input from the user

kilometers = float(input("Enter value in kilometers: "))

# conversion factor

conv\_fac = 0.621371

# calculate miles

miles = kilometers \* conv\_fac

print('%0.2f kilometers is equal to %0.2f miles' %(kilometers,miles))

# Python program to check if the input number is odd or even.

# A number is even if division by 2 gives a remainder of 0.

# If the remainder is 1, it is an odd number.

num = int(input("Enter a number: "))

if (num % 2) == 0:

print("{0} is Even".format(num))

else:

print("{0} is Odd".format(num))

# Python program to check if year is a leap year or not

year = 2000

# To get year (integer input) from the user

# year = int(input("Enter a year: "))

if (year % 4) == 0:

if (year % 100) == 0:

if (year % 400) == 0:

print("{0} is a leap year".format(year))

else:

print("{0} is not a leap year".format(year))

else:

print("{0} is a leap year".format(year))

else:

print("{0} is not a leap year".format(year))

# Python program to find the largest number among the three input numbers

# change the values of num1, num2 and num3

# for a different result

num1 = 10

num2 = 14

num3 = 12

# uncomment following lines to take three numbers from user

#num1 = float(input("Enter first number: "))

#num2 = float(input("Enter second number: "))

#num3 = float(input("Enter third number: "))

if (num1 >= num2) and (num1 >= num3):

largest = num1

elif (num2 >= num1) and (num2 >= num3):

largest = num2

else:

largest = num3

print("The largest number is", largest)

**# Program to check if a number is prime or not**

num = 407

# To take input from the user

#num = int(input("Enter a number: "))

# prime numbers are greater than 1

if num > 1:

# check for factors

for i in range(2,num):

if (num % i) == 0:

print(num,"is not a prime number")

print(i,"times",num//i,"is",num)

break

else:

print(num,"is a prime number")

# if input number is less than

# or equal to 1, it is not prime

else:

print(num,"is not a prime number")

# Program to check Armstrong numbers in a certain interval

lower = 100

upper = 2000

for num in range(lower, upper + 1):

# order of number

order = len(str(num))

# initialize sum

sum = 0

temp = num

while temp > 0:

digit = temp % 10

sum += digit \*\* order

temp //= 10

if num == sum:

print(num)

# Python Program to find the factors of a number

# This function computes the factor of the argument passed

def print\_factors(x):

print("The factors of",x,"are:")

for i in range(1, x + 1):

if x % i == 0:

print(i)

num = 320 print\_factors(num)

# Program make a simple calculator

# This function adds two numbers

def add(x, y):

return x + y

# This function subtracts two numbers

def subtract(x, y):

return x - y

# This function multiplies two numbers

def multiply(x, y):

return x \* y

# This function divides two numbers

def divide(x, y):

return x / y

print("Select operation.")

print("1.Add")

print("2.Subtract")

print("3.Multiply")

print("4.Divide")

# Take input from the user

choice = input("Enter choice(1/2/3/4): ")

num1 = float(input("Enter first number: "))

num2 = float(input("Enter second number: "))

if choice == '1':

print(num1,"+",num2,"=", add(num1,num2))

elif choice == '2':

print(num1,"-",num2,"=", subtract(num1,num2))

elif choice == '3':

print(num1,"\*",num2,"=", multiply(num1,num2))

elif choice == '4':

print(num1,"/",num2,"=", divide(num1,num2))

else:

print("Invalid input")

In Python, we can implement a matrix as a nested list (list inside a list). We can treat each element as a row of the matrix.

For example X = [[1, 2], [4, 5], [3, 6]] would represent a 3x2 matrix. The first row can be selected as X[0]. And, the element in the first-row first column can be selected as X[0][0].

Transpose of a matrix is the interchanging of rows and columns. It is denoted as X'. The element at ith row and jth column in X will be placed at jth row and ith column in X'. So if X is a 3x2 matrix, X' will be a 2x3 matrix.

Here are a couple of ways to accomplish this in Python.

**Matrix Transpose using Nested Loop**

# Program to transpose a matrix using a nested loop

X = [[12,7],

[4 ,5],

[3 ,8]]

result = [[0,0,0],

[0,0,0]]

# iterate through rows

for i in range(len(X)):

# iterate through columns

for j in range(len(X[0])):

result[j][i] = X[i][j]

for r in result:

print(r)

Run Code

**Output**

[12, 4, 3]

[7, 5, 8]

In this program, we have used nested for loops to iterate through each row and each column. At each point we place the X[i][j] element into result[j][i].

**Matrix Transpose using Nested List Comprehension**

''' Program to transpose a matrix using list comprehension'''

X = [[12,7],

[4 ,5],

[3 ,8]]

result = [[X[j][i] for j in range(len(X))] for i in range(len(X[0]))]

for r in result:

print(r)

Run Code

The output of this program is the same as above. We have used nested list comprehension to iterate through each element in the matrix.

When we want to send the same invitations to many people, the body of the mail does not change. Only the name (and maybe address) needs to be changed.

Mail merge is a process of doing this. Instead of writing each mail separately, we have a template for body of the mail and a list of names that we merge together to form all the mails.

## Source Code to Merge Mails

# Python program to mail merger

# Names are in the file names.txt

# Body of the mail is in body.txt

# open names.txt for reading

with open("names.txt",'r',encoding = 'utf-8') as names\_file:

# open body.txt for reading

with open("body.txt",'r',encoding = 'utf-8') as body\_file:

# read entire content of the body

body = body\_file.read()

# iterate over names

for name in names\_file:

mail = "Hello "+name+body

# write the mails to individual files

with open(name.strip()+".txt",'w',encoding = 'utf-8') as mail\_file:

mail\_file.write(mail)

For this program, we have written all the names in separate lines in the file "names.txt". The body is in the "body.txt" file.

We open both the files in reading mode and iterate over each name using a for loop. A new file with the name "[name].txt" is created, where name is the name of that person.

We use strip() method to clean up leading and trailing whitespaces (reading a line from the file also reads the newline '\n' character). Finally, we write the content of the mail into this file using the write() method.

JPEG (pronounced "jay-peg") stands for Joint Photographic Experts Group. It is one of the most widely used compression techniques for image compression.

Most of the file formats have headers (initial few bytes) which contain useful information about the file.

For example, jpeg headers contain information like height, width, number of color (grayscale or RGB) etc. In this program, we find the resolution of a jpeg image reading these headers, without using any external library.

**Source Code of Find Resolution of JPEG Image**

def jpeg\_res(filename):

""""This function prints the resolution of the jpeg image file passed into it"""

# open image for reading in binary mode

with open(filename,'rb') as img\_file:

# height of image (in 2 bytes) is at 164th position

img\_file.seek(163)

# read the 2 bytes

a = img\_file.read(2)

# calculate height

height = (a[0] << 8) + a[1]

# next 2 bytes is width

a = img\_file.read(2)

# calculate width

width = (a[0] << 8) + a[1]

print("The resolution of the image is",width,"x",height)

jpeg\_res("img1.jpg")

**Output**

The resolution of the image is 280 x 280

In this program, we opened the image in binary mode. Non-text files must be open in this mode. The height of the image is at 164th position followed by width of the image. Both are 2 bytes long.

Note that this is true only for JPEG File Interchange Format (JFIF) standard. If your image is encode using other standard (like EXIF), the code will not work.

We convert the 2 bytes into a number using bitwise shifting operator <<. Finally, the resolution is displayed.