# LAB # 1 Tasks

Step 1: Prompt users for input

Step 2 : Adding operators

Step 3: Adding conditional statements

Step 4: Defining functions

Step 5: Improving the code

# Program: Calculator.py

Step 1: number\_1 = int(input('Enter your first number: '))

number\_2 = int(input('Enter your second number: '))

Step 2: Adding operators

number\_1 = int(input('Enter your first number: '))

number\_2 = int(input('Enter your second number: '))

print(number\_1 + number\_2)

We want the user to receive confirmation about the numbers they are entering and the operator that is being used alongside the produced result. To do this, we’ll be using [string formatters](https://www.digitalocean.com/community/tutorials/how-to-use-string-formatters-in-python-3) to help us properly format our text and provide feedback.

number\_1 = int(input('Enter your first number: '))

number\_2 = int(input('Enter your second number: '))

print('{} + {} = '.format(number\_1, number\_2))

print(number\_1 + number\_2)

At this point, you can add the rest of the operators to the program with the same format we have used for addition:

Step 3: Conditional Statements

we want the user to be able to choose among the different operators

We’ll write a string on a few different lines by using triple quotes:

We are using each of the operator symbols for users to make their choice, so if the user wants division to be performed, they will type /. We could choose whatever symbols we want, though, like 1 for addition, or b for subtraction.

Because we are asking users for input, we want to use the input() function. We’ll put the string inside of the input() function, and pass the value of that input to a variable, which we’ll name operation.

operation = input('''

Please type in the math operation you would like to complete:

+ for addition

- for subtraction

\* for multiplication

/ for division

''')

At this point, if we run our program it doesn’t matter what we input at the first prompt, so let’s add our conditional statements into the program. Because of how we have structured our program, the ifstatement will be where the addition is performed, there will be 3 else-if or elif statements for each of the other operators, and the else statement will be put in place to handle an error if the person did not input an operator symbol.

operation = input('''

Please type in the math operation you would like to complete:

+ for addition

- for subtraction

\* for multiplication

/ for division

''')

number\_1 = int(input('Enter your first number: '))

number\_2 = int(input('Enter your second number: '))

**if** operation == '+':

print('{} + {} = '.format(number\_1, number\_2))

print(number\_1 + number\_2)

**elif** operation == '-':

print('{} - {} = '.format(number\_1, number\_2))

print(number\_1 - number\_2)

**elif** operation == '\*':

print('{} \* {} = '.format(number\_1, number\_2))

print(number\_1 \* number\_2)

**elif** operation == '/':

print('{} / {} = '.format(number\_1, number\_2))

print(number\_1 / number\_2)

**else**:

print('You have not typed a valid operator, please run the program again.')

Step 4: Functions

To handle the ability to perform the program as many times as the user wants, we’ll define some functions. Let’s first put our existing code block into a function. We’ll name the function calculate() and add an additional layer of indentation within the function itself.

# Define our function

**defcalculate**():

operation = input('''

Please type in the math operation you would like to complete:

+ for addition

- for subtraction

\* for multiplication

/ for division

''')

number\_1 = int(input('Please enter the first number: '))

number\_2 = int(input('Please enter the second number: '))

**if** operation == '+':

print('{} + {} = '.format(number\_1, number\_2))

print(number\_1 + number\_2)

**elif** operation == '-':

print('{} - {} = '.format(number\_1, number\_2))

print(number\_1 - number\_2)

**elif** operation == '\*':

print('{} \* {} = '.format(number\_1, number\_2))

print(number\_1 \* number\_2)

**elif** operation == '/':

print('{} / {} = '.format(number\_1, number\_2))

print(number\_1 / number\_2)

**else**:

print('You have not typed a valid operator, please run the program again.')

# Call calculate() outside of the function

calculate()

Next, let’s create a second function made up of more conditional statements. In this block of code, we want to give the user the choice as to whether they want to calculate again or not. We can base this off of our calculator conditional statements, but in this case we’ll only have one if, one elif, and one else to handle errors.

We’ll name this function again(), and add it below our def calculate(): code block.

# Define again() function to ask user if they want to use the calculator again

**defagain**():

# Take input from user

calc\_again = input('''

Do you want to calculate again?

Please type Y for YES or N for NO.

''')

# If user types Y, run the calculate() function

**if**calc\_again == 'Y':

calculate()

# If user types N, say good-bye to the user and end the program

**elif**calc\_again == 'N':

print('See you later.')

# If user types another key, run the function again

**else**:

again()

# Call calculate() outside of the function

calculate()

Although there is some error-handling with the else statement above, we could probably do a little better to accept, say, a lower-case y and n in addition to the upper-case Y and N. To do that, let’s add the [string function](https://www.digitalocean.com/community/tutorials/an-introduction-to-string-methods-in-python-3) str.upper():

...

**defagain**():

calc\_again = input('''

Do you want to calculate again?

Please type Y for YES or N for NO.

''')

# Accept 'y' or 'Y' by adding str.upper()

**if**calc\_again.upper() == 'Y':

calculate()

# Accept 'n' or 'N' by adding str.upper()

**elif**calc\_again.upper() == 'N':

print('See you later.')

**else**:

again()

...

At this point, we should add the again() function to the end of the calculate() function so that we can trigger the code that asks the user whether or not they would like to continue.

………………

……………….

………………..

## Step 5 — Improving the code

You can add a welcome function, for example, that welcomes people to the program at the top of the program’s code

Q1. Write a script that asks a user for a number. The script adds 3 to that number. Then multiplies the result by 2, subtracts 4, subtracts twice the original number, adds 3, then prints the result.

print(number +3)\*2-4-(number \*2)+3

Q2. Write a script that takes input as radius then calculate area of circle. (hint: A= πr²).

from math import pi

r=float(input("Enter radius value :"))  
area = (pi\*r\*\*2)  
print(area)

Q3. Write a Python program to calculate the sum of three given numbers, if the values are equal then return thrice of their sum.

defsum\_thrice(x, y, z):

sum= x + y + z

if x == y == z:

sum=sum\*3

returnsum

print(sum\_thrice(1,2,3))

print(sum\_thrice(3,3,3))

Q4. Write a function which takes three number a, b and c, the task is that we have to find the greatest element in among in given number.

Q5. Write a function to convert date format. Given a time in 12-hour AM/PM format, convert it to military (24-hour) time.

**Note :** Midnight is 12:00:00 AM on a 12-hour clock and 00:00:00 on a 24-hour clock. Noon is 12:00:00 PM on 12-hour clock and 12:00:00 on 24-hour clock.

Examples :

Input : 11:21:30 PM

Output : 23:21:30

Input : 12:12:20 AM

Output : 00:12:20