```
# pdb
import pdb

num1=(input("Enter first number: "))
num2=(input("Enter second number: "))

pdb.set_trace()

try:
  result=num1-num2
except:
  result=int(num1)-int(num2)
print(result)
```

```
Enter first number: 12
Enter second number: 10


PYDEV DEBUGGER WARNING:
sys.settrace() should not be used when the debugger is being used.
This may cause the debugger to stop working correctly.
If this is needed, please check:
http://pydev.blogspot.com/2007/06/why-cant-pydev-debugger-work-
with.html
to see how to restore the debug tracing back correctly.
Call Location:
  File "/usr/lib/python3.10/bdb.py", line 336, in set_trace
    sys.settrace(self.trace_dispatch)


--Return--
None
> <ipython-input-1-78e2904c2a94>(7)<cell line: 7>()
      5 num2=(input("Enter second number: "))
      6
----> 7 pdb.set_trace()
      8
      9 try:

ipdb> result=num1-num2
*** TypeError: unsupported operand type(s) for -: 'str' and 'str'
ipdb> result=int(num1)-int(num2)
ipdb> result
2
ipdb> exit


PYDEV DEBUGGER WARNING:
sys.settrace() should not be used when the debugger is being used.
This may cause the debugger to stop working correctly.
If this is needed, please check:
```

```
http://pydev.blogspot.com/2007/06/why-cant-pydev-debugger-work-
with.html
to see how to restore the debug tracing back correctly.
Call Location:
  File "/usr/lib/python3.10/bdb.py", line 361, in set_quit
    sys.settrace(None)


def add(a,b):
  return a+b

def sub(a,b):
  return a-b

def mul(a,b):
  return a*b

def div(a,b):
  try a//b
  except zerodivisionerror:
    print("cannot divide by zero")
  return a/b

  def cal():
    operator=input("Enter operator: ")
    a=int(input("Enter first number: "))
    b=int(input("Enter second number: "))
    match operator:
      case "+":
        print(add(a,b))
      case "-":
        print(sub(a,b))
      case "*":
        print(mul(a,b))
      case "/":
        print(div(a,b))

#OOPS
class animal:
  head= True
  def legs():
    print("There are 4 legs")
  def tail():
    print("There is a tail")

class Dog:
  head= True
  def legs():
    print("Dogs have 4 legs")
  def tail():
```

```
    print("Dogs has a tail")

class Cat:
  head= True
  def legs():
    print("Cats have 4 legs")
  def tail():
    print("Cats has a tail")

animal().head
animal.legs()
animal.tail()

Dog().head
Dog.legs()
Dog.tail()

Cat().head
Cat.legs()
Cat.tail()
```

```
There are 4 legs
There is a tail
Dogs have 4 legs
Dogs has a tail
Cats have 4 legs
Cats has a tail
```

```
#classes
import math
class arithmetic:
  def __init__(self,val1:int,val2:int) ->None:
    self.val1=val1
    self.val2=val2

  def add(self)->float:
    return self.val1+self.val2
  def sub(self)->float:
    return self.val1-self.val2
  def mul(self)->float:
    return self.val1*self.val2
  def div(self)->float:
    try:
      self.val1/self.val2
    except zerodivisionerror:
      print("Cannot divide by zero")
    return  self.val1/self.val2

class scientific:
  def __init__(self,val:int) ->None:
```

```
    self.val=val

  def sine(self):
    return math.sin(math.radians(self.val))
  def cosine(self):
    return math.cos(math.radians(self.val))

def main():
  choice=input("Press 1. for arithmetic and 2. for scientific")
  match choice:
    case "1":
      a=int(input("Enter First Number"))
      b=int(input("Enter Second Number"))
      arth:arithmetic=arithmetic(a,b) #arth=arithmetic(a,b)
      oper=input("1. Add 2. Sub 3. Mul 4. Div")
      if oper=="1": print(arth.add())
      elif oper=="2": print(arth.sub())
      elif oper=="3": print(arth.mul())
      elif oper=="4": print(arth.div())
      else: print("Invalid choice")

    case "2":
      ang=int(input("Enter the angle"))
      sci:scientific=scientific(ang)
      oper=input("1. sin 2. cos")
      if oper=="1": print(sci.sine())
      elif oper=="2": print(sci.cosine())
      else: print("Invalid")
    case _:
      print("Invalid Choice")


if __name__=="__main__":
  main()



Press 1. for arithmetic and 2. for scientific1
Enter First Number12
Enter Second Number13
1. Add 2. Sub 3. Mul 4. Div1
25
```