Basic Projects

Mad Libs Generator Python Game

Madlib is a game where someone fills in random words such as adjectives, nouns, a person's name, etc., without knowing the context of the story. As a result, the final story usually sounds funny, absurd, or unreasonable. I made this program in Python and developed it in PyCharm. When the program starts, users will be presented with a GUI that has 6 story options that can be played. When the user clicks one of the options, the user will be asked to input various words depending on the prompts. After that, the story will appear in the console using the words the user entered earlier.

In the program I made, I fixed several things such as correcting typos, removing unnecessary words, adding more story options, and changing the GUI colors to make it more interesting. I also changed the concept of the button placement with `pack()` to make it simpler and easier to manage.

Code (Programmed in Python and developed in PyCharm):

```
# import module
from tkinter import *

# initialize window
root = Tk()
root.geometry('400x600')
root.title('DataFlair-Mad Libs Generator')
root.configure(bg='#FDE2E4')

# Header labels
Label(root, text='Mad Libs Generator \n Have Fun!', font='arial 20 bold',
bg='#FDE2E4', fg='black').pack(pady=(10,5))
Label(root, text='Click Any One :', font='arial 15 bold', bg='#FDE2E4',
fg='black').pack(pady=(10))
Label(root, text='Daniel Daniello | 2025-12-01', font='arial 15 bold',
fg='blue', bg='#FDE2E4').pack(pady=5)


################Stories##############

def madlib1(): # The Photographer
    animals = input('enter a animal name : ')
    profession = input('enter a profession name: ')
    cloth = input('enter a piece of cloth name: ')
    things = input('enter a thing name: ')
    name = input('enter a name: ')
    place = input('enter a place name: ')
    verb = input('enter a verb in ing form: ')
    food = input('food name: ')
    print(
        'say ' + food + ', the photographer said as the camera flashed! ' +
name + ' and I had gone to ' + place + ' to get our photos taken today. The
first photo we really wanted was a picture of us dressed as ' + animals + '
pretending to be a ' + profession + ' .when we saw the second photo, it was
exactly what I wanted. We both looked like ' + things + ' wearing ' + cloth
+ ' and ' + verb + ' --exactly what I had in mind')
```

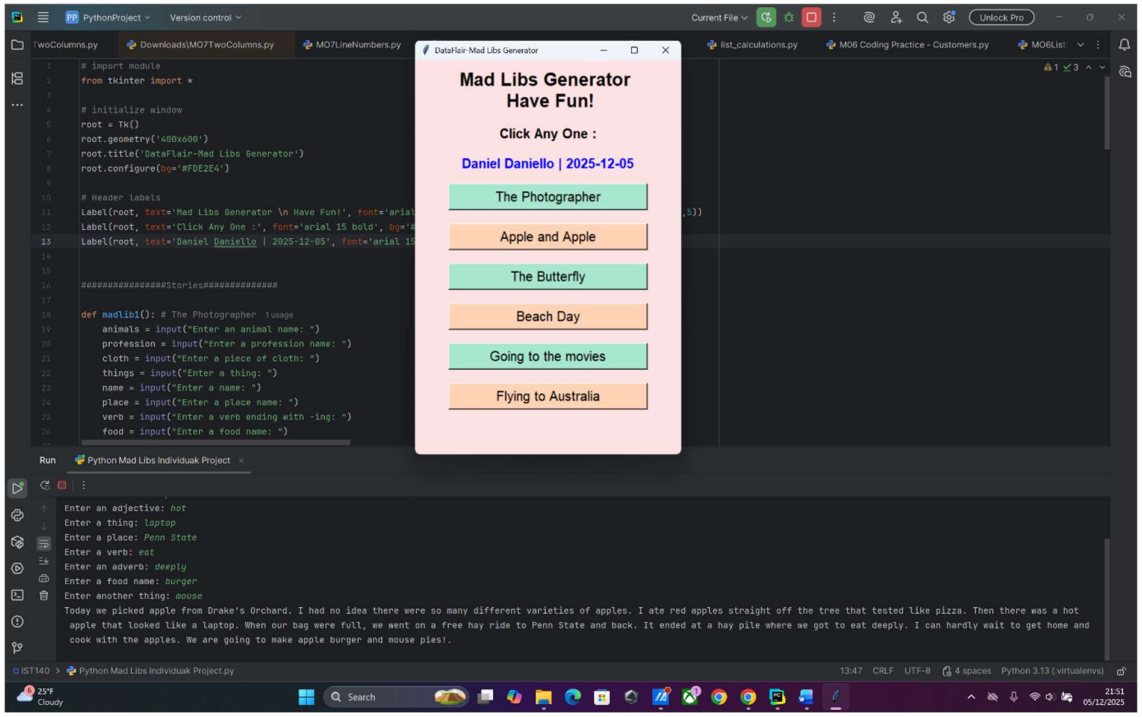
```
def madlib2(): # The Butterfly
    adjective = input('enter adjective : ')
    color = input('enter a color name : ')
    thing = input('enter a thing name :')
    place = input('enter a place name : ')
    person = input('enter a person name : ')
    adjective1 = input('enter a adjective : ')
    insect = input('enter a insect name : ')
    food = input('enter a food name : ')
    verb = input('enter a verb name : ')

    print(
        'Last night I dreamed I was a ' + adjective + ' butterfly with ' +
color + ' splocthes that looked like ' + thing + ' .I flew to ' + place + '
with my bestfriend and ' + person + ' who was a ' + adjective1 + ' ' +
insect + ' .We ate some ' + food + ' when we got there and then decided to
' + verb + ' and the dream ended when I said-- lets ' + verb + '.')


def madlib3(): # Apple and Apple
    person = input('enter person name: ')
    color = input('enter color : ')
    foods = input('enter food name : ')
    adjective = input('enter aa adjective name: ')
    thing = input('enter a thing name : ')
    place = input('enter place : ')
    verb = input('enter verb : ')
    adverb = input('enter adverb : ')
    food = input('enter food name: ')
    things = input('enter a thing name : ')

    print(
        'Today we picked apple from ' + person + "'s Orchard. I had no idea
there were so many different varieties of apples. I ate " + color + '
apples straight off the tree that tested like ' + foods + '. Then there was
a ' + adjective + ' apple that looked like a ' + thing + '. When our bag
were full, we went on a free hay ride to ' + place + ' and back. It ended
at a hay pile where we got to ' + verb + ' ' + adverb + '. I can hardly
wait to get home and cook with the apples. We are going to make appple ' +
food + ' and ' + things + ' pies!.')

def madlib4(): # Beach day
    place = input("Enter a place name: ")
    person = input("Enter a person name: ")
    adjective1 = input("Enter an adjective: ")
    noun1 = input("Enter a noun: ")
    noun2 = input("Enter another noun: ")
    adjective2 = input("Enter another adjective: ")
    adjective3 = input("Enter another adjective: ")
    noun3 = input("Enter another noun: ")
    number = input("Enter a number: ")
    noun4 = input("Enter another noun: ")
    food = input("Enter a food name: ")
    verb = input("Enter a verb: ")

    print("I went to the beach in " + place + " today with " + person + ".
We had a " + adjective1 + " time. We first built a " + noun1 + " and then
we ran around in the " + noun2 + " for a little while. The water was a bit
" + adjective2 + ". The sun was very " + adjective3 + " so we made sure to
```

```
wear a lot of " + noun3 + ". We spent " + number + " hours there. At the
end of the day, we treated ourselves to a " + noun4 + " and had " + food +
" for dinner. I definitely want to go to the beach again but next time I
want to " + verb + ".")

def madlib5(): # Going to the movies
    person1 = input("Enter a person name: ")
    person2 = input("Enter another person name: ")
    movie = input("Enter a movie name: ")
    adjective1 = input("Enter an adjective: ")
    verb1 = input("Enter a verb: ")
    noun1 = input("Enter a noun: ")
    candy = input("Enter a candy name: ")
    food = input("Enter a food name: ")
    verb2 = input("Enter a verb: ")
    verb3 = input("Enter another verb: ")
    verb4 = input("Enter another verb: ")
    adjective2 = input("Enter another adjective: ")
    adjective3 = input("Enter another adjective: ")
    adjective4 = input("Enter another adjective: ")
    adjective5 = input("Enter another adjective: ")

    print("I went to the movies yesterday with " + person1 + " and " +
person2 + " We saw " + movie + ". It was " + adjective1 + ". At one part, I
even " + verb1 + " and ran for the " + noun1 + ". During the move, we ate "
+ candy + " and " + food + ". I got mad because the person sitting behind
me kept " + verb2 + " during the movie and wouldn't stop " + verb3 + ". He
was asked to leave after he " + verb4 + " across the theatre. It was pretty
" + adjective2 + ". Overall, I liked the movie because it was " +
adjective3 + " and the main character was super " + adjective4 + ".
Hopefully next time the people sitting behind me will be more " +
adjective5 + ".")

def madlib6(): # Flying to Australia
    noun1 = input("Enter a noun: ")
    number = input("Enter a number: ")
    noun2 = input("Enter another noun: ")
    food = input("Enter a food name: ")
    adjective1 = input("Enter an adjective: ")
    verb1 = input("Enter a verb: ")
    work = input("Enter a type of work/job: ")
    noun3 = input("Enter another noun: ")
    adjective2 = input("Enter another adjective: ")
    verb2 = input("Enter a verb: ")
    verb3 = input("Enter another verb: ")
    adjective3 = input("Enter another adjective: ")

    print("I decided to go on a vacation to Australia with " + noun1 + ".
We got to the airport " + number + " hours early. When we went through
security, I got stopped because I forgot to take " + noun2 + " out of my
pocket. We got some " + food + " for the flight and arrived at the gate.
Once we boarded the plane, I was sitting next to a very " + adjective1 + "
man. He spent the entire flight " + verb1 + " and talking about his job
doing " + work + ". Whenever I tried to sleep, he would step around me to
go to the " + noun3 + ". I was so " + adjective2 + ". Since I couldn't
sleep, I decided to " + verb2 + " and " + verb3 + " instead. Finally, we
arrived in Australia. All in all, the flight was " + adjective3 + ".")


# Buttons to select stories
Button(root, text='The Photographer', font='arial 15', command=madlib1,
```

```
bg='#A8E6CF', fg='black').pack(pady=10, fill='x', padx=50)
Button(root, text='Apple and Apple', font='arial 15', command=madlib3,
bg='#FFD3B6', fg='black').pack(pady=10, fill='x', padx=50)
Button(root, text='The Butterfly', font='arial 15', command=madlib2,
bg='#A8E6CF', fg='black').pack(pady=10, fill='x', padx=50)
Button(root, text='Beach Day', font='arial 15', command=madlib4,
bg='#FFD3B6', fg='black').pack(pady=10, fill='x', padx=50)
Button(root, text='Going to the movies', font='arial 15', command=madlib5,
bg='#A8E6CF', fg='black').pack(pady=10, fill='x', padx=50)
Button(root, text='Flying to Australia', font='arial 15', command=madlib6,
bg='#FFD3B6', fg='black').pack(pady=10, fill='x', padx=50)



root.mainloop()
```

Output:

```
Enter a person name: Drake
Enter a color: red
Enter a food name: pizza
Enter an adjective: hot
Enter a thing: laptop
Enter a place: Penn State
Enter a verb: eat
Enter an adverb: deeply
Enter a food name: burger
Enter another thing: mouse
Today we picked apple from Drake's Orchard. I had no idea there were so
many different varieties of apples. I ate red apples straight off the tree
that tested like pizza. Then there was a hot apple that looked like a
laptop. When our bag were full, we went on a free hay ride to Penn State
and back. It ended at a hay pile where we got to eat deeply. I can hardly
wait to get home and cook with the apples. We are going to make apple
burger and mouse pies!.
```



Source Code:

```
#import module
from tkinter
import *

# initialize window
root = Tk()
root.geometry('300x300')
root.title('DataFlair-Mad Libs Generator')
Label(root, text= 'Mad Libs Generator \n Have Fun!' , font = 'arial 20
bold').pack()
Label(root, text = 'Click Any One :', font = 'arial 15 bold').place(x=40,
y=80)


################Stories##############

def madlib1():

    animals= input('enter a animal name : ')
    profession = input('enter a profession name: ')
    cloth = input('enter a piece of cloth name: ')
    things = input('enter a thing name: ')
    name= input('enter a name: ')
    place = input('enter a place name: ')
    verb = input('enter a verb in ing form: ')
    food = input('food name: ')
    print('say ' + food + ', the photographer said as the camera flashed! '
+ name + ' and I had gone to ' + place +' to get our photos taken today.
The first photo we really wanted was a picture of us dressed as ' + animals
+ ' pretending to be a ' + profession + ' .when we saw the second photo, it
was exactly what I wanted. We both looked like ' + things + ' wearing ' +
cloth + ' and ' + verb + ' --exactly what I had in mind')



def madlib2():

    adjactive = input('enter adjective : ')
    color = input('enter a color name : ')
    thing = input('enter a thing name :')
    place = input('enter a place name : ')
    person= input('enter a person name : ')
    adjactive1 = input('enter a adjactive : ')
    insect= input('enter a insect name : ')
    food = input('enter a food name : ')
    verb = input('enter a verb name : ')

    print('Last night I dreamed I was a ' +adjactive+ ' butterfly with ' +
color+ ' splocthes that looked like '+thing+ ' .I flew to ' + place+ ' with
my bestfriend and '+person+ ' who was a '+adjactive1+ ' ' +insect +' .We
ate some ' +food+ ' when we got there and then decided to '+verb+ ' and the
dream ended when I said-- lets ' +verb+ '.')



def madlib3():
    person = input('enter person name: ')
```

```
    color = input('enter color : ')
    foods = input('enter food name : ')
    adjective = input('enter aa adjective name: ')
    thing = input('enter a thing name : ')
    place = input('enter place : ')
    verb = input('enter verb : ')
    adverb = input('enter adverb : ')
    food = input('enter food name: ')
    things = input('enter a thing name : ')


    print('Today we picked apple from '+person+ "'s Orchard. I had no idea
there were so many different varieties of apples. I ate " +color+ ' apples
straight off the tree that tested like '+foods+ '. Then there was a
'+adjective+ ' apple that looked like a ' + thing + '.When our bag were
full, we went on a free hay ride to '+place+ ' and back. It ended at a hay
pile where we got to ' +verb+ ' ' +adverb+ '. I can hardly wait to get home
and cook with the apples. We are going to make appple '+food+ ' and
'+things+' pies!.')



######buttons
Button(root, text= 'The Photographer', font ='arial 15', command= madlib1,
bg = 'ghost white').place(x=60, y=120)
Button(root, text= 'apple and apple', font ='arial 15', command = madlib3 ,
bg = 'ghost white').place(x=70, y=180)
Button(root, text= 'The Butterfly', font ='arial 15', command = madlib2, bg
= 'ghost white').place(x=80, y=240)

root.mainloop()
```

Intermediate Projects

Calculator in Python

This program is a common calculator project. This calculator has the functions of adding, subtracting, division, and multiplications, including special operators such as power, factorial, pi, brackets. In the calculator that I made, I improved many things, especially the large number of bugs in the original code.

I also improved the visual appearance of the buttons and the display entry, changed the color to make it more attractive, and adjusted the GUI layout so that each row and column in the grid can expand evenly. The bugs I fixed include issues with the error message, the “C” button, the “=” button, the input and output behavior, and several other minor problems. Simply put, this program is more robust and is free of the previous bugs.

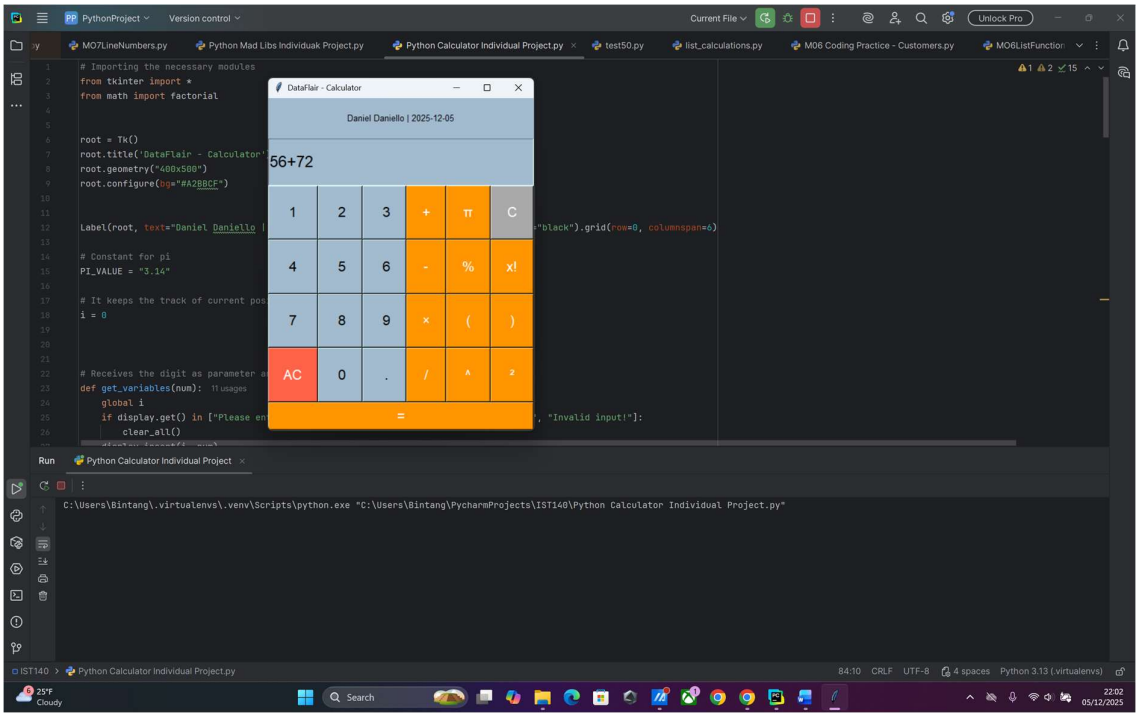
Code (Programmed in Python and developed in PyCharm):

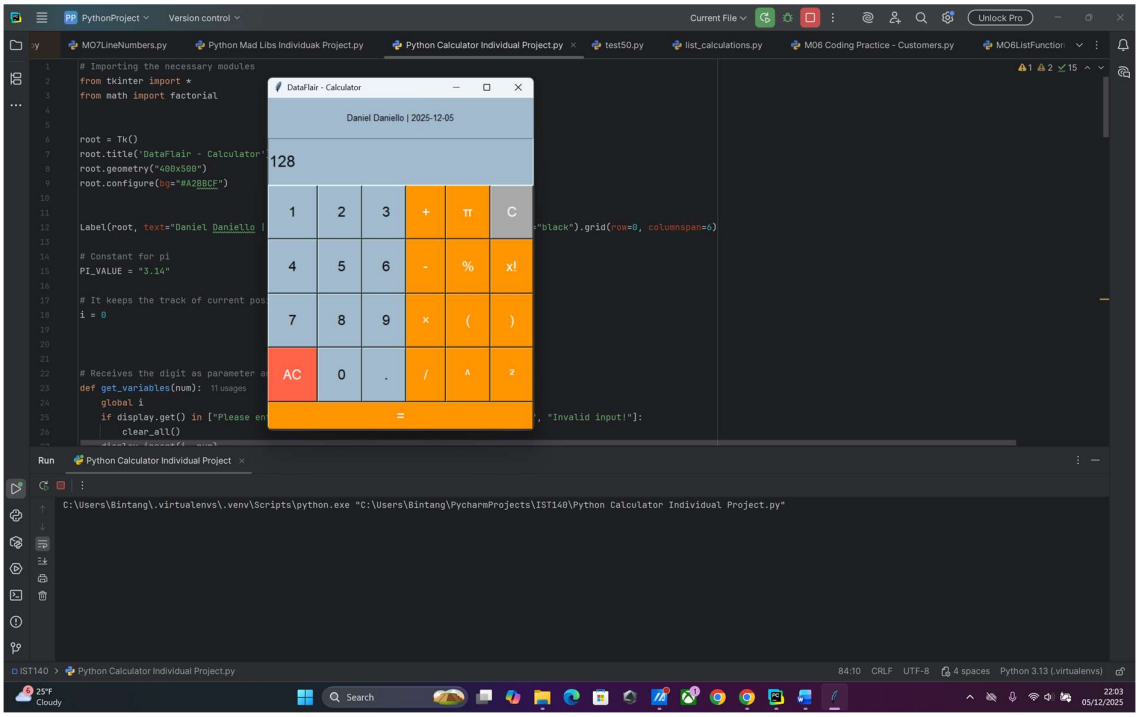
```
# Importing the necessary modules
from tkinter import *
from math import factorial


root = Tk()
root.title('DataFlair - Calculator')
root.geometry("400x500")
root.configure(bg="#A2BBCF")

Label(root, text="Daniel Daniello | 2025-12-05", font=("Arial", 10),
bg="#A2BBCF", fg="black").grid(row=0, columnspan=6)

# Constant for pi
PI_VALUE = "3.14"

# It keeps the track of current position on the input text field
i = 0

# Receives the digit as parameter and display it on the input field
def get_variables(num):
    global i
    if display.get() in ["Please enter numbers to calculate!", "Error in
calculation!", "Invalid input!"]:
        clear_all()
    display.insert(i, num)
    i = display.index(END)


# Calculate function scans the string to evaluates and display it
def calculate():
    global i
    entire_string = display.get()
    valid_chars = "0123456789.+-*/%^()π²×"
    if any(ch not in valid_chars for ch in entire_string):
        clear_all()
        display.insert(0, "Invalid input!")
```

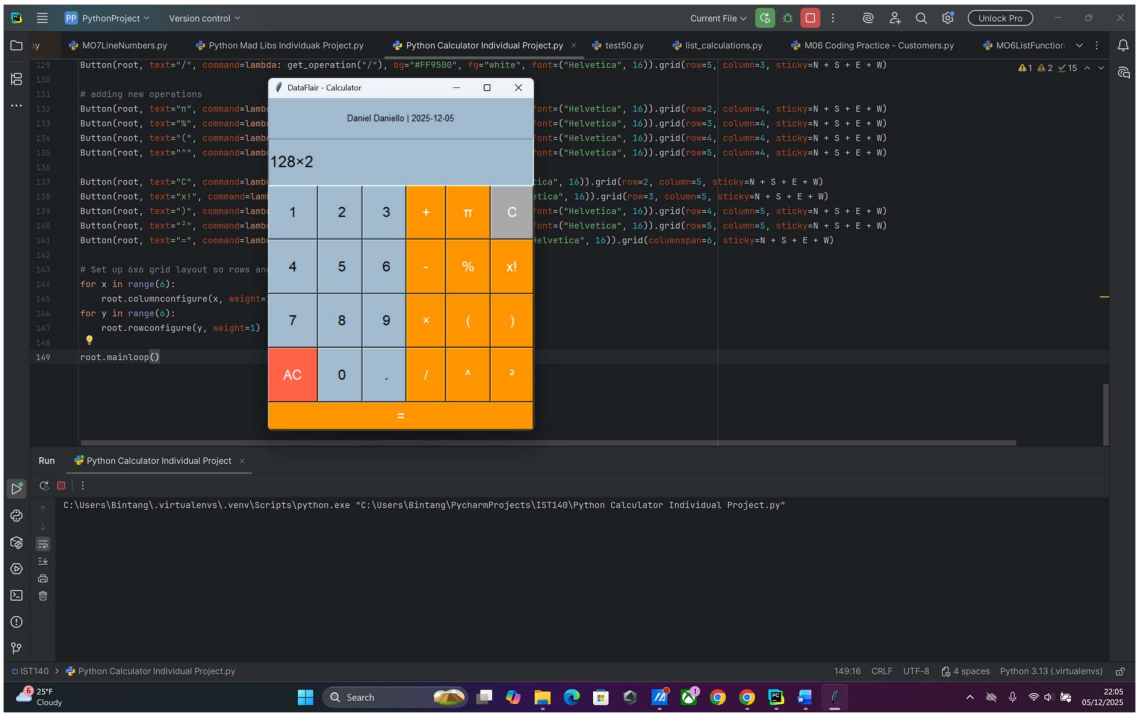
```
        i = 0
        return

    try:
        entire_string = entire_string.replace("π", "*3.14") if not
entire_string.startswith("π") else entire_string.replace("π", "3.14",
1).replace("π", "*3.14")
        entire_string = entire_string.replace("²", "**2")
        entire_string = entire_string.replace("^", "**")
        entire_string = entire_string.replace("%", "/100")
        entire_string = entire_string.replace("×", "*")
        a = entire_string
        result = eval(a)
        clear_all()
        display.insert(0, result)
        i = display.index(END)
    except Exception:
        clear_all()
        display.insert(0, "Error in calculation!")
        i = 0


# Function which takes operator as input and displays it on the input field
def get_operation(operator):
    global i
    if display.get() in ["Please enter numbers to calculate!", "Error in
calculation!", "Invalid input!"]:
        clear_all()
    display.insert(i, operator)
    i = display.index(END)


# Function to clear the input field
def clear_all():
    global i
    display.delete(0, END)
    i = 0


# Function which works like backspace
def undo():
    entire_string = display.get()
    if len(entire_string):
        new_string = entire_string[:-1]
        clear_all()
        display.insert(0, new_string)
        global i
        = display.index(END)
    else:
        clear_all()
        i = 0


# Function to calculate the factorial and display it
def fact():
    global i
    entire_string = display.get()
    try:
        result = factorial(int(entire_string))
        clear_all()
        display.insert(0, result)
```

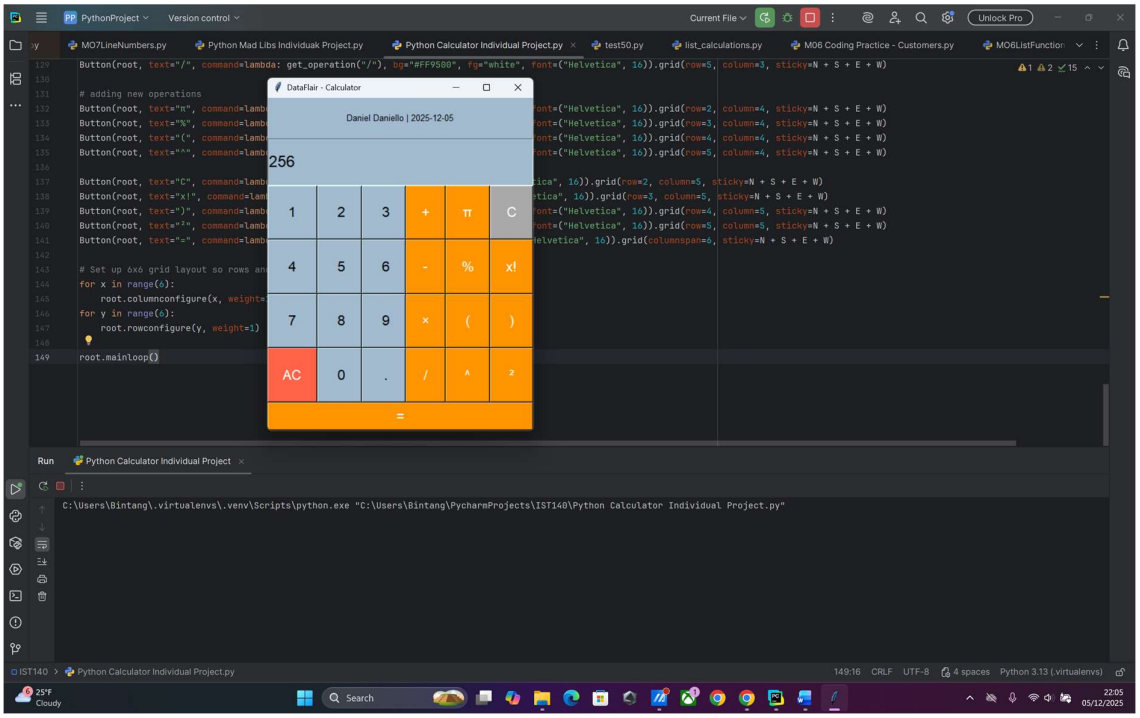
```
        i = display.index(END)
    except Exception:
        clear_all()
        display.insert(0, "Please enter numbers to calculate!")
        i = display.index(END)


# -------------------------------------UI Design ------------------------
-------------------

# adding the input field
display = Entry(root, font=("Helvetica", 18), bg="#A2BBCF", fg="black")
display.grid(row=1, columnspan=6, sticky=N + E + W + S)

# Code to add buttons to the Calculator

Button(root, text="1", command=lambda: get_variables(1), bg="#A2BBCF",
fg="black", font=("Helvetica", 16)).grid(row=2, column=0, sticky=N + S + E
+ W)
Button(root, text=" 2", command=lambda: get_variables(2), bg="#A2BBCF",
fg="black", font=("Helvetica", 16)).grid(row=2, column=1, sticky=N + S + E
+ W)
Button(root, text=" 3", command=lambda: get_variables(3), bg="#A2BBCF",
fg="black", font=("Helvetica", 16)).grid(row=2, column=2, sticky=N + S + E
+ W)

Button(root, text="4", command=lambda: get_variables(4), bg="#A2BBCF",
fg="black", font=("Helvetica", 16)).grid(row=3, column=0, sticky=N + S + E
+ W)
Button(root, text=" 5", command=lambda: get_variables(5), bg="#A2BBCF",
fg="black", font=("Helvetica", 16)).grid(row=3, column=1, sticky=N + S + E
+ W)
Button(root, text=" 6", command=lambda: get_variables(6), bg="#A2BBCF",
fg="black", font=("Helvetica", 16)).grid(row=3, column=2, sticky=N + S + E
+ W)

Button(root, text="7", command=lambda: get_variables(7), bg="#A2BBCF",
fg="black", font=("Helvetica", 16)).grid(row=4, column=0, sticky=N + S + E
+ W)
Button(root, text=" 8", command=lambda: get_variables(8), bg="#A2BBCF",
fg="black", font=("Helvetica", 16)).grid(row=4, column=1, sticky=N + S + E
+ W)
Button(root, text=" 9", command=lambda: get_variables(9), bg="#A2BBCF",
fg="black", font=("Helvetica", 16)).grid(row=4, column=2, sticky=N + S + E
+ W)

# adding other buttons to the calculator
Button(root, text="AC", command=lambda: clear_all(), bg="#FF6347",
fg="white", font=("Helvetica", 16)).grid(row=5, column=0, sticky=N + S + E
+ W)
Button(root, text=" 0", command=lambda: get_variables(0), bg="#A2BBCF",
fg="black", font=("Helvetica", 16)).grid(row=5, column=1, sticky=N + S + E
+ W)
Button(root, text=" .", command=lambda: get_variables("."), bg="#A2BBCF",
fg="black", font=("Helvetica", 16)).grid(row=5, column=2, sticky=N + S + E
+ W)

Button(root, text="+", command=lambda: get_operation("+"), bg="#FF9500",
fg="white", font=("Helvetica", 16)).grid(row=2, column=3, sticky=N + S + E
+ W)
Button(root, text="-", command=lambda: get_operation("-"), bg="#FF9500",
```

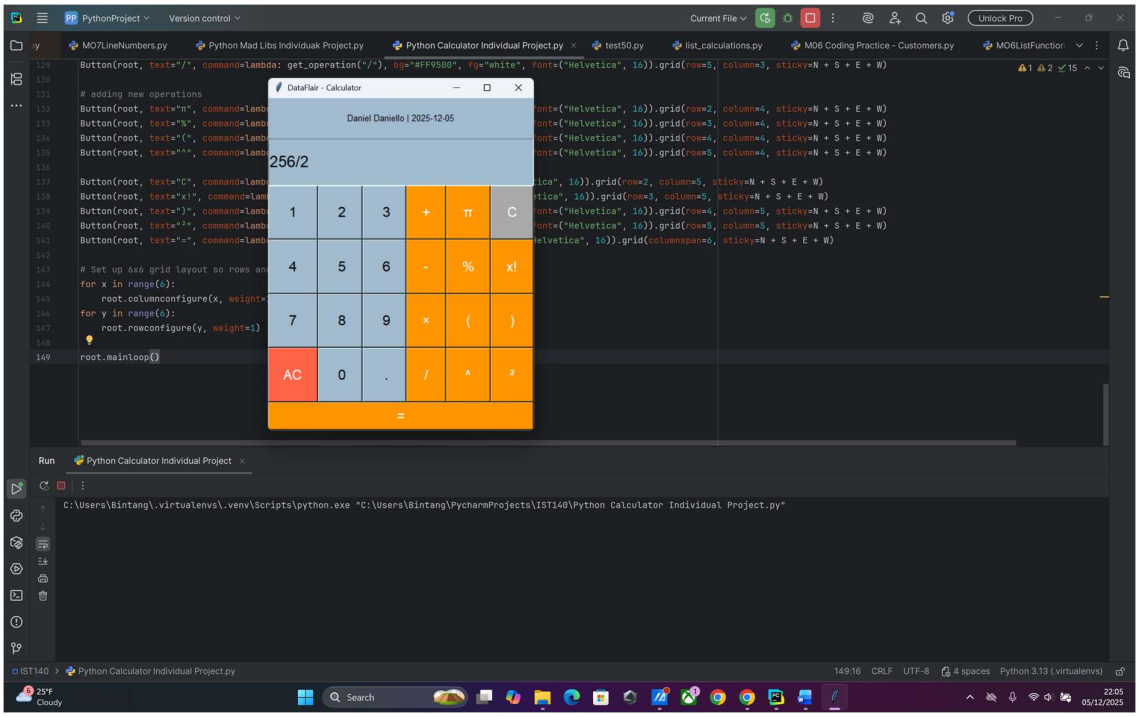
```
fg="white", font=("Helvetica", 16)).grid(row=3, column=3, sticky=N + S + E
+ W)
Button(root, text="×", command=lambda: get_operation("×"), bg="#FF9500",
fg="white", font=("Helvetica", 16)).grid(row=4, column=3, sticky=N + S + E
+ W)
Button(root, text="/", command=lambda: get_operation("/"), bg="#FF9500",
fg="white", font=("Helvetica", 16)).grid(row=5, column=3, sticky=N + S + E
+ W)

# adding new operations
Button(root, text="π", command=lambda: get_operation("π"), bg="#FF9500",
fg="white", font=("Helvetica", 16)).grid(row=2, column=4, sticky=N + S + E
+ W)
Button(root, text="%", command=lambda: get_operation("%"), bg="#FF9500",
fg="white", font=("Helvetica", 16)).grid(row=3, column=4, sticky=N + S + E
+ W)
Button(root, text="(", command=lambda: get_operation("("), bg="#FF9500",
fg="white", font=("Helvetica", 16)).grid(row=4, column=4, sticky=N + S + E
+ W)
Button(root, text="^", command=lambda: get_operation("^"), bg="#FF9500",
fg="white", font=("Helvetica", 16)).grid(row=5, column=4, sticky=N + S + E
+ W)

Button(root, text="C", command=lambda: undo(), bg="#A9A9A9", fg="white",
font=("Helvetica", 16)).grid(row=2, column=5, sticky=N + S + E + W)
Button(root, text="x!", command=lambda: fact(), bg="#FF9500", fg="white",
font=("Helvetica", 16)).grid(row=3, column=5, sticky=N + S + E + W)
Button(root, text=")", command=lambda: get_operation(")"), bg="#FF9500",
fg="white", font=("Helvetica", 16)).grid(row=4, column=5, sticky=N + S + E
+ W)
Button(root, text="²", command=lambda: get_operation("²"), bg="#FF9500",
fg="white", font=("Helvetica", 16)).grid(row=5, column=5, sticky=N + S + E
+ W)
Button(root, text="=", command=lambda: calculate(), bg="#FF9500",
fg="white", font=("Helvetica", 16)).grid(columnspan=6, sticky=N + S + E +
W)

# Set up 6x6 grid layout so rows and columns expand equally
for x in range(6):
    root.columnconfigure(x, weight=1)
for y in range(6):
    root.rowconfigure(y, weight=1)

root.mainloop()
```

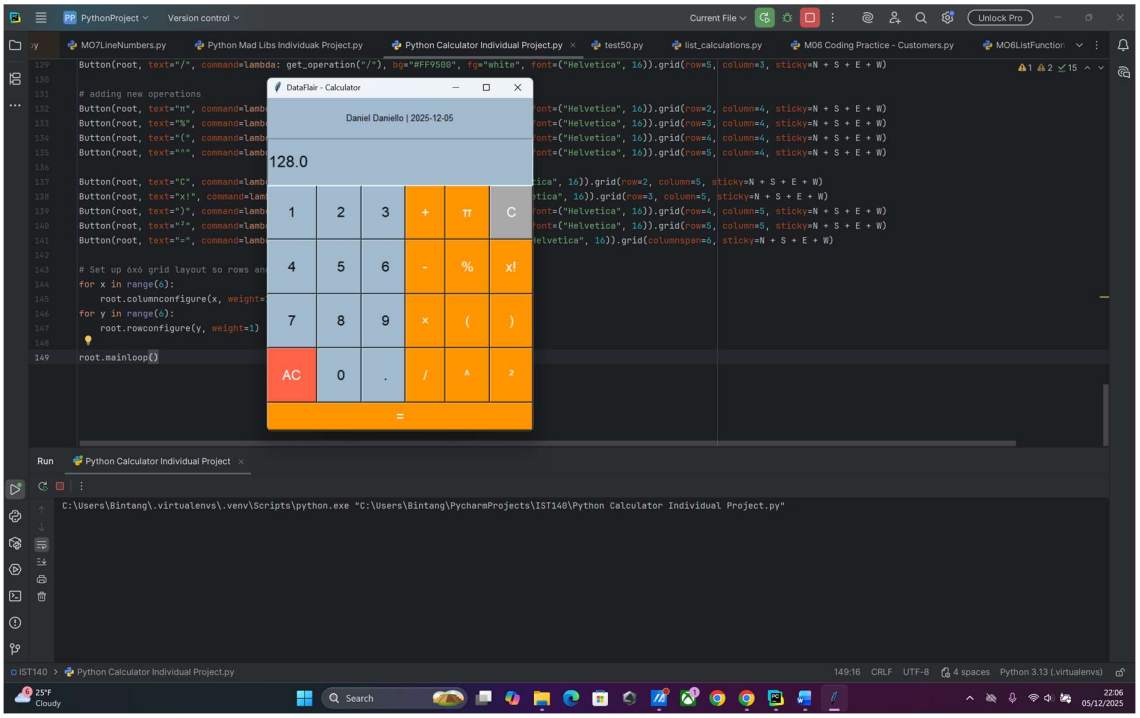

DataFlair - Calculator
Daniel Daniello | 2025-12-05
56+72
1 2 3 + π C
4 5 6 - % x!
7 8 9 × ( )
AC 0 . / ^ ²
=
C:\Users\Bintang\.virtualenvs\.venv\Scripts\python.exe "C:\Users\Bintang\PycharmProjects\IST140\Python Calculator Individual Project.py"

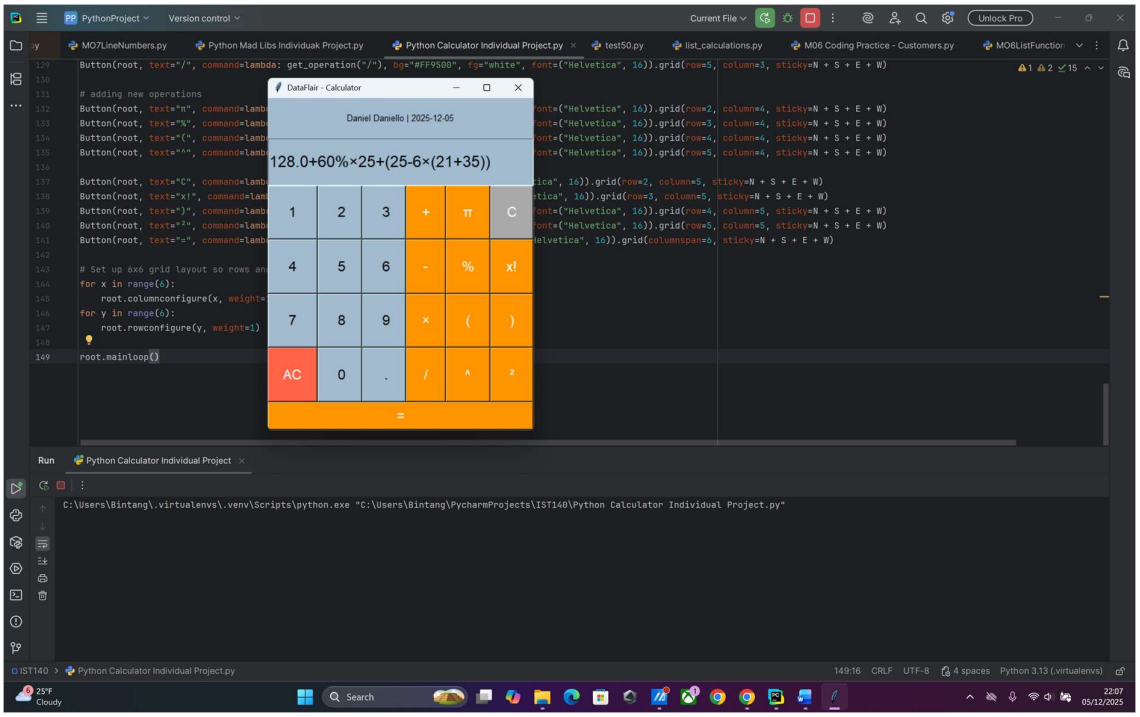

DataFlair - Calculator
Daniel Daniello | 2025-12-05
128
1 2 3 + π C
4 5 6 - % x!
7 8 9 × ( )
AC 0 . / ^ ²
=
C:\Users\Bintang\.virtualenvs\.venv\Scripts\python.exe "C:\Users\Bintang\PycharmProjects\IST140\Python Calculator Individual Project.py"

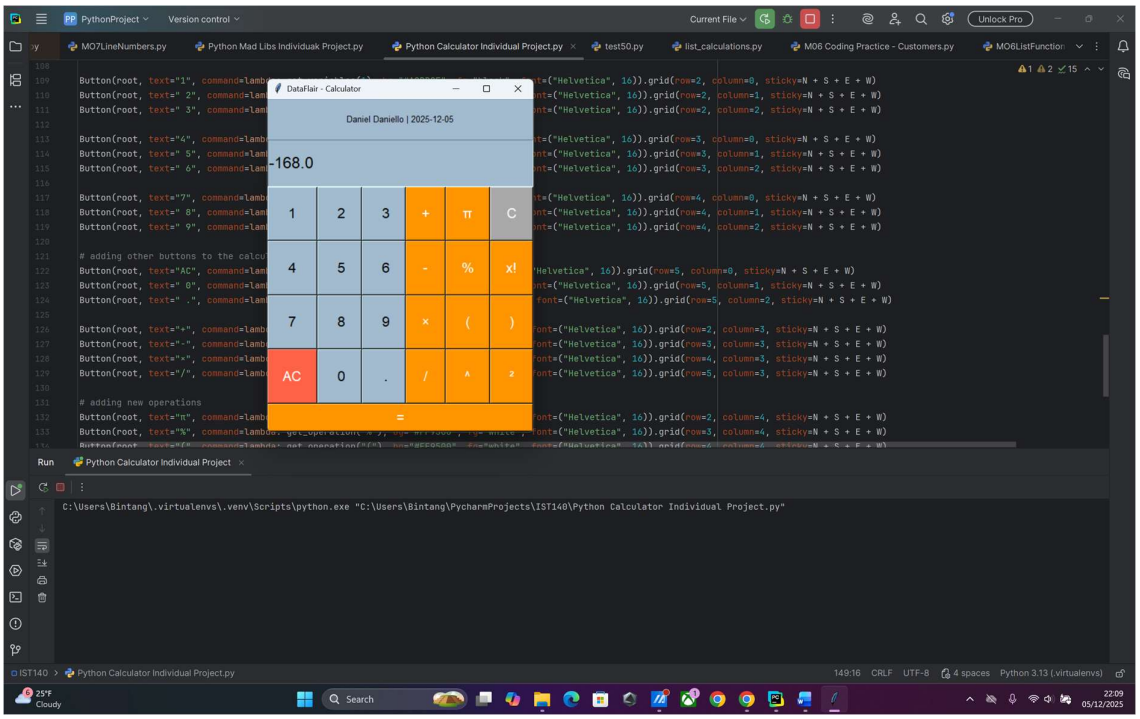

DataFlair - Calculator
Daniel Daniello | 2025-12-05
128×2
1 2 3 + π C
4 5 6 - % x!
7 8 9 × ( )
AC 0 . / ^ ²
=


DataFlair - Calculator
Daniel Daniello | 2025-12-05
256
1 2 3 + π C
4 5 6 - % x!
7 8 9 × ( )
AC 0 . / ^ ²
=


DataFlair - Calculator
Daniel Daniello | 2025-12-05
256/2
1 2 3 + π C
4 5 6 - % x!
7 8 9 × ( )
AC 0 . / ^ ²
=
C:\Users\Bintang\.virtualenvs\.venv\Scripts\python.exe "C:\Users\Bintang\PycharmProjects\IST140\Python Calculator Individual Project.py"


DataFlair - Calculator
Daniel Daniello | 2025-12-05
128.0
1 2 3 + π C
4 5 6 - % x!
7 8 9 × ( )
AC 0 . / ^ ²
=
C:\Users\Bintang\.virtualenvs\.venv\Scripts\python.exe "C:\Users\Bintang\PycharmProjects\IST140\Python Calculator Individual Project.py"





Source code:

```
#Importing the necessary modules
from tkinter import *
from math import factorial

root = Tk()
root.title('DataFlair - Calculator')
```

```
#It keeps the track of current position on the input text field
i = 0
# Receives the digit as parameter and display it on the input field
def get_variables(num):
    global i
    display.insert(i,num)
    i+=1

# Calculate function scans the string to evaluates and display it
def calculate():
    entire_string = display.get()
    try:
        a = entire_string
        result = eval(a)
        clear_all()
        display.insert(0,result)
    except Exception:
        clear_all()
        display.insert(0,"Error")

# Function which takes operator as input and displays it on the input field
def get_operation(operator):
    global i
    length = len(operator)
    display.insert(i,operator)
    i+=length

#Function to clear the input field
def clear_all():
    display.delete(0,END)

#Function which works like backspace
def undo():
    entire_string = display.get()
    if len(entire_string):
        new_string = entire_string[:-1]
        clear_all()
        display.insert(0,new_string)
    else:
        clear_all()
        display.insert(0,"Error")

#Function to calculate the factorial and display it
def fact():
    entire_string = display.get()
    try:
        result = factorial(int(entire_string))
        clear_all()
        display.insert(0,result)
    except Exception:
        clear_all()
        display.insert(0,"Error")

#-------------------------------------UI Design ------------------------
------------------

#adding the input field
display = Entry(root)
display.grid(row=1,columnspan=6,sticky=N+E+W+S)

#Code to add buttons to the Calculator
```

```
Button(root,text="1",command = lambda
:get_variables(1)).grid(row=2,column=0, sticky=N+S+E+W)
Button(root,text=" 2",command = lambda
:get_variables(2)).grid(row=2,column=1, sticky=N+S+E+W)
Button(root,text=" 3",command = lambda
:get_variables(3)).grid(row=2,column=2, sticky=N+S+E+W)

Button(root,text="4",command = lambda
:get_variables(4)).grid(row=3,column=0, sticky=N+S+E+W)
Button(root,text=" 5",command = lambda
:get_variables(5)).grid(row=3,column=1, sticky=N+S+E+W)
Button(root,text=" 6",command = lambda
:get_variables(6)).grid(row=3,column=2, sticky=N+S+E+W)

Button(root,text="7",command = lambda
:get_variables(7)).grid(row=4,column=0, sticky=N+S+E+W)
Button(root,text=" 8",command = lambda
:get_variables(8)).grid(row=4,column=1, sticky=N+S+E+W)
Button(root,text=" 9",command = lambda
:get_variables(9)).grid(row=4,column=2, sticky=N+S+E+W)

#adding other buttons to the calculator
Button(root,text="AC",command=lambda :clear_all()).grid(row=5,column=0,
sticky=N+S+E+W)
Button(root,text=" 0",command = lambda
:get_variables(0)).grid(row=5,column=1, sticky=N+S+E+W)
Button(root,text=" .",command=lambda :get_variables(".")).grid(row=5,
column=2, sticky=N+S+E+W)


Button(root,text="+",command= lambda
:get_operation("+")).grid(row=2,column=3, sticky=N+S+E+W)
Button(root,text="-",command= lambda :get_operation("-
")).grid(row=3,column=3, sticky=N+S+E+W)
Button(root,text="*",command= lambda
:get_operation("*")).grid(row=4,column=3, sticky=N+S+E+W)
Button(root,text="/",command= lambda
:get_operation("/")).grid(row=5,column=3, sticky=N+S+E+W)

# adding new operations
Button(root,text="pi",command= lambda
:get_operation("*3.14")).grid(row=2,column=4, sticky=N+S+E+W)
Button(root,text="%",command= lambda
:get_operation("%")).grid(row=3,column=4, sticky=N+S+E+W)
Button(root,text="(",command= lambda
:get_operation("(")).grid(row=4,column=4, sticky=N+S+E+W)
Button(root,text="exp",command= lambda
:get_operation("**")).grid(row=5,column=4, sticky=N+S+E+W)

Button(root,text="<-",command= lambda :undo()).grid(row=2,column=5,
sticky=N+S+E+W)
Button(root,text="x!", command= lambda: fact()).grid(row=3,column=5,
sticky=N+S+E+W)
Button(root,text=")",command= lambda
:get_operation(")")).grid(row=4,column=5, sticky=N+S+E+W)
Button(root,text="^2",command= lambda
:get_operation("**2")).grid(row=5,column=5, sticky=N+S+E+W)
Button(root,text="^2",command= lambda
:get_operation("**2")).grid(row=5,column=5, sticky=N+S+E+W)
```

```
Button(root,text="=",command= lambda :calculate()).grid(columnspan=6,
sticky=N+S+E+W)


root.mainloop()
```