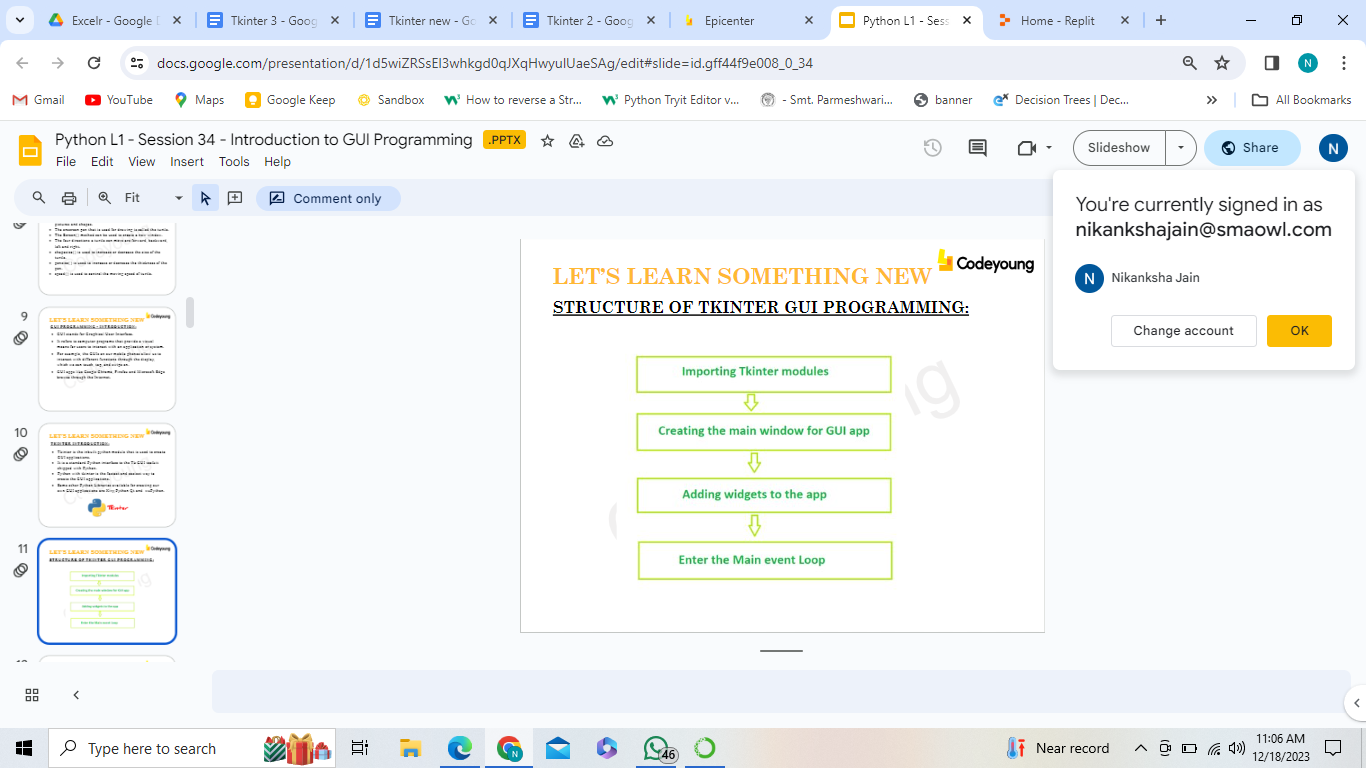
**GUI PROGRAMMING - INTRODUCTION:**

* GUI stands for Graphical User Interface.
* It refers to computer programs that provide a visual means for users to interact with an application or system.
* For example, the GUIs on our mobile phones allow us to interact with different functions through the display, which we can touch, tap, and swipe on.
* GUI apps like Google Chrome, Firefox and Microsoft Edge browse through the Internet.​

**TKINTER INTRODUCTION:**

* Tkinter is the inbuilt python module that is used to create GUI applications.
* It is a standard Python interface to the Tk GUI toolkit shipped with Python.
* Python with tkinter is the fastest and easiest way to create the GUI applications.
* Some other Python Libraries available for creating our own GUI applications are​ Kivy​,Python Qt​ and wxPython.​



**METHODS USED:**

* The two main methods used with Tkinter GUI areTk() and mainloop( ).

1. Tk( ) method is used to create a main window.

**Tk(screenName=None, baseName=None, className=’Tk’)**

* The basic code used to create the main window is:  
  **m=tkinter.Tk()** where m is the name of the main window object.
* The parameters passed are optional.

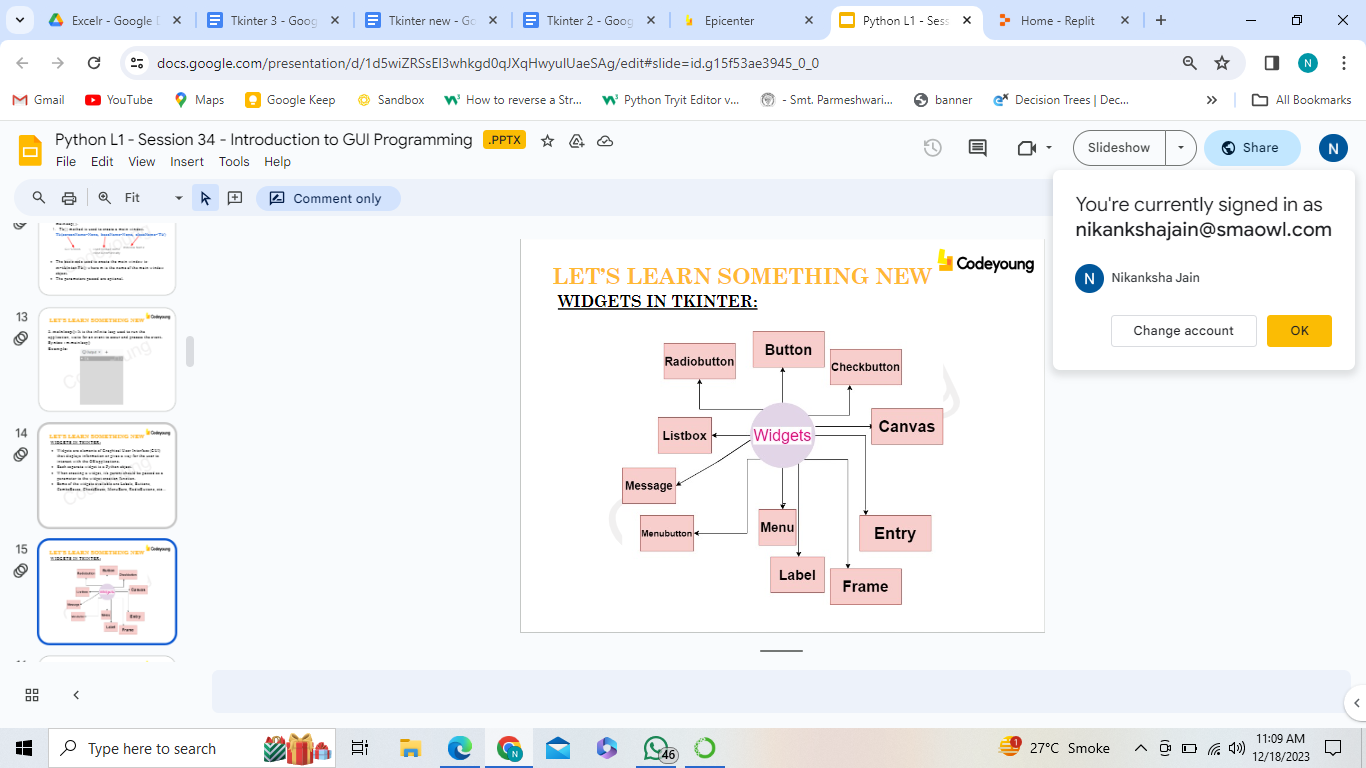
**2. mainloop():** It is the infinite loop used to run the application, waits for an event to occur and process the event.

**Syntax :** m.mainloop()

**Example:**

**WIDGETS IN TKINTER:**

* Widgets are elements of Graphical User Interface (GUI) that displays information or gives a way for the user to interact with the OS/applications.
* Each separate widget is a Python object.
* When creating a widget, it’s parent should be passed as a parameter to the widget creation function.
* Some of the widgets available are Labels, Buttons, ComboBoxes, CheckBoxes, MenuBars, RadioButtons, etc…

****

**LABEL WIDGET:**

* Label widget is used to insert display boxes where text or images can be placed.
* The text displayed by this widget can be changed any time.
* Label widget is used to underline the part of the text and span the text across multiple lines.
* A label can use only one font at a time to display text.
* To use a label, we need to specify what to display in it (this can be text, a bitmap, or an image).​
* **Syntax:** ​w=Label(master, option=value)
* **NOTE:** master is the parameter used to represent the parent window.

**OPTIONAL ARGUMENTS FOR LABEL:**

Options are passed as parameters separated by commas.

Some of them are:

* **anchor:** This option is used to control the positioning of the text. The default is anchor=CENTER, which centers the text in the available space.
* **bg:**This option is used to set the normal background colour displayed behind the label and indicator.
* **height:**This option is used to set the vertical dimension of the new frame.
* **width:**Width of the label in characters. If this option is not set, the label will be sized to fit its contents.
* **bd:**This option is used to set the size of the border around the indicator. Default bd value is set on 2 pixels.

**Before the Tasks**

* *Tony Stark wants to build a program that looks good*
* *He doesn’t want to use the same text based input system*
* *He wants to create a modern app that has buttons,windows etc.*
* *Can you help Tony in his journey to build a modern app using Python?*

**Task 1**

**Task 1: Importing it right.**

* *So what is a library?*[Wait for the learner to respond]
* *Yes exactly, a library is a collection of books and you can pick up any book from there and read it. Right?*[Learner responds with a yes]
* *Tkinter is a library in Python that allows us to build Python based Apps.*
* *It adds a Graphical User Interface to it. This Interface will allow us to add Buttons, texts and other elements to our program*
* *Now with click of a button, we can do things in our program*
* *Inorder to use Tkinter in our Python Program, we need to import it.*from tkinter import \*
* *Great! Now we have the power of Tkinter*
* *In Order to use our components, we need to create a Window.*
* *Window will be like a house to everything that we do in Tkinter*window = Tk()
* *Let’s give it a name using the window.title command*window.title(“Hello world”)
* *Let’s keep the window in a loop so that it stays on the screen*window.mainloop()
* *Great! Now let’s run the program and see if we get an output or not*

**Task 1:Solution**

from tkinter import \*

window = Tk()

window.title("Hello world")

window.mainloop()

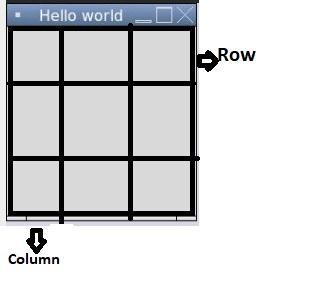
**Output Displayed:**

****

**Task 2**

* This widget is used to add buttons in the application.
* The buttons can display text or images that convey the purpose of the buttons.
* Function or a method can be attached to a button which is called automatically when the button is clicked.
* **Syntax:** w = Button( master, option=value, ... )

**Task 2: Buttoning it!**

* *Mr. Stark is now happy to see the progress*
* *Now we have a nice window that has the title of hello world*
* *Inorder to add a button to the window, we will use Button Widget*
* *Widget is small functional parts of your program that can be joined together in your program*
* *Let’s add a button to our program*btn=Button(window,text="Click me")
* *Here we have added a variable called btn and used Button widget in that****window refers to the window that will house our button  
  text= is the text that will be written on our button***
* *Great!   
  Now in order to find our button in the window, we need to place it inside a grid.*
* *A grid looks like this* [Teachers to share their screen and show]  
  
* *Let’s add our button to the grid*btn.grid(column=1, row=0)

**Task 2: Solution**

from tkinter import \*

window = Tk()

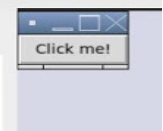
window.title("Hello world")

btn=Button(window,text="Click me!")

btn.grid(column=1, row=0)

window.mainloop()

**Output Displayed:**

****

**Task 3**

**Task 3: Texting it**

* *Now let’s add a text with the button.*
* *Let’s add a text label below the button*
* *A Label is a widget that is used to add text to the window*lbl = Label(window, text="Hello,Let's Code")
* *Now let’s place it in the grid*lbl.grid(column=2, row=0)

**Task 3: Solution**

from tkinter import \*

window = Tk()

window.title("Hello world")

btn=Button(window,text="Click me!")

btn.grid(column=1, row=0)

lbl = Label(window, text="Hello,Let's Code")

lbl.grid(column=2, row=0)

window.mainloop()

**Output Displayed:**

****

* *Awesome. Now we have a button and a label in our app.*
* *Now finally let’s make our app large*window.geometry('400x400')
* *Geometry function will allow us to change the size of our window, make it large or small*
* *The values that we entered are the length and breadth of the window.*

**Output Displayed:**

****

**Task 4**

**Task 4: Do it on your own**

* *Mr.Stark is very happy with your progress.*
* *He now wants you to add one more label that says “Learning with Tkinter” in the next line*
* *And another button beside it that says “Click me again”*
* *Can you do it on your own without any help?*
* *Hints: Understand the grid, if you want to bring it to the next line, what value should be assigned to the row?*
* *You need to add two more variables (One for label, one for button)*[Push the learners to try it on their own and find out the solution]

**Task 4: Solution***[Probable Solution]*

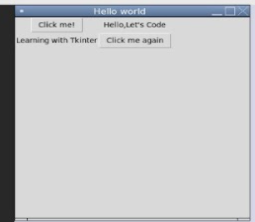
lbl2=Label(window, text="Learning with Tkinter")

lbl2.grid(column=1, row=1)

btn2=Button(window, text="Click me again")

btn2.grid(column=2, row=1)

**Output Displayed:**

****

**In the end**

* *You learnt how to use the label and button widget in Tkinter and understood the grid structure as well.*

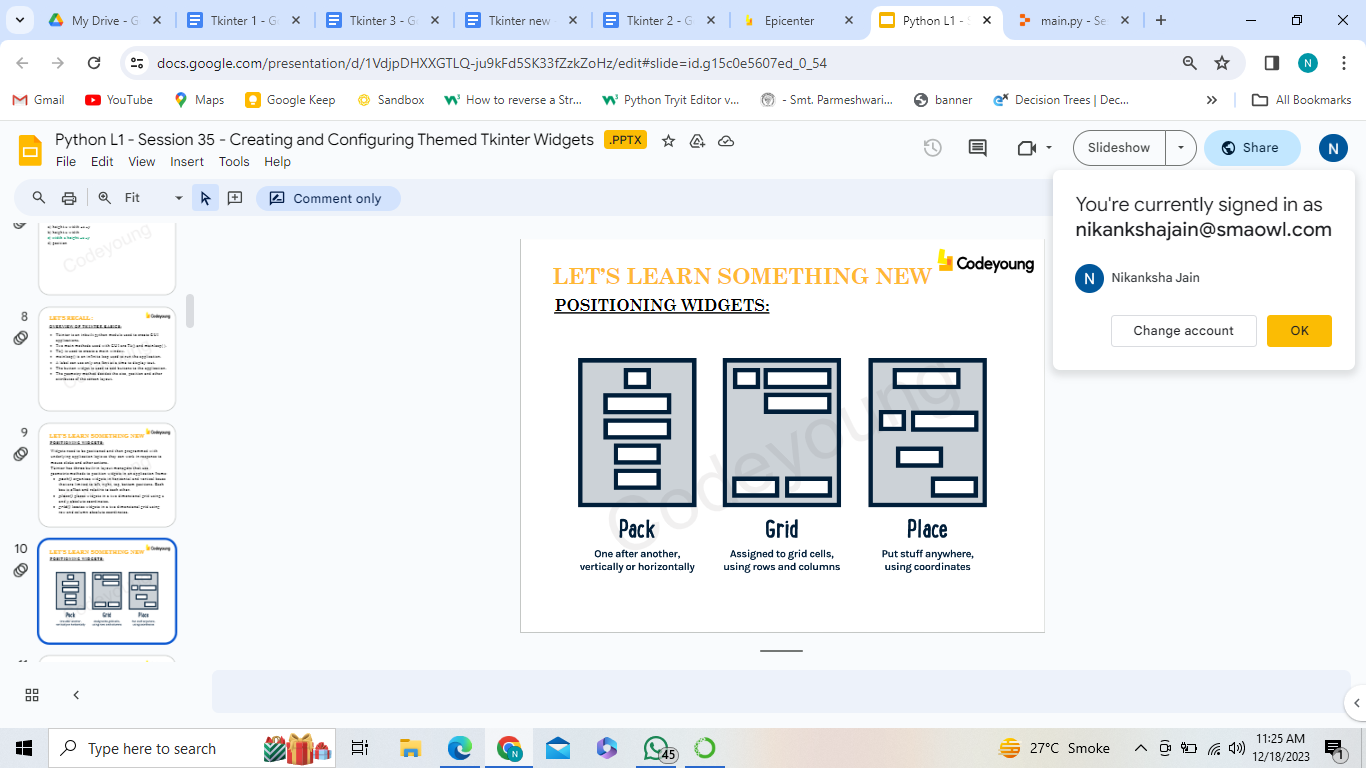
**Solution**

* Link to Solution: <https://repl.it/@SurajSharma12/Tkinter-Start-Solution>

**POSITIONING WIDGETS**

Widgets need to be positioned and then programmed with underlying application logic so they can work in response to mouse clicks and other actions.

Tkinter has **three** built-in layout managers that use geometric methods to position widgets in an application frame:

* ***pack()***organizes widgets in horizontal and vertical boxes that are limited to left, right, top, bottom positions. Each box is offset and relative to each other.
* ***place()***places widgets in a two dimensional grid using x and y absolute coordinates.
* ***grid()***locates widgets in a two dimensional grid using row and column absolute coordinates.
* 
* *pack()* declares the positioning of widgets in relation to each other instead of declaring the precise location.
* *pack(*) is limited in precision compared to *place()* and *grid()* which feature absolute positioning.
* For simple positioning of widgets vertically or horizontally in relation to each other, *pack()* layout manager is used.

***pack()* has four padding options:**

* Internal padding
* External padding
* *Padx*, which pads along the x axis
* *Pady*, which pads along the y axis

**Syntax:** widget.pack( pack\_options )

A list of possible options that can be passed in to pack() are:

* **expand:** If the expand is set to true, the widget expands to fill any space.
* **Fill:** By default, the fill is set to NONE. However, we can set it to X or Y to determine whether the widget contains any extra space.
* **size:** It represents the size of the parent to which the widget is to be placed on the window.

import tkinter as tk

window = tk.Tk()

window.title("Hello world")

window.geometry("300x300")

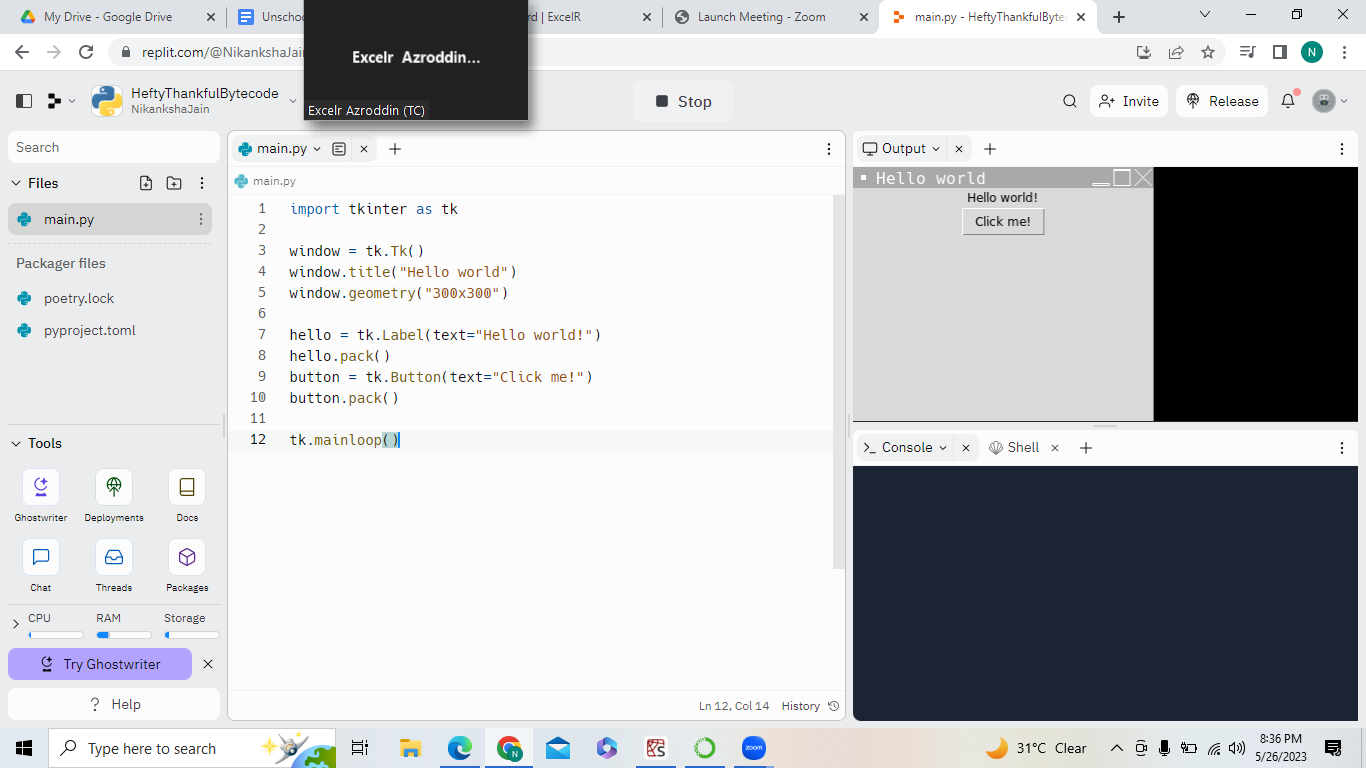
hello = tk.Label(text="Hello world!")

hello.pack()

button = tk.Button(text="Click me!")

button.pack()

tk.mainloop()



**Place function:**

from tkinter import \*

window = Tk()

window.title("Hello world")

btn=Button(window,text="Click me!")

btn.place(x=150,y=100)

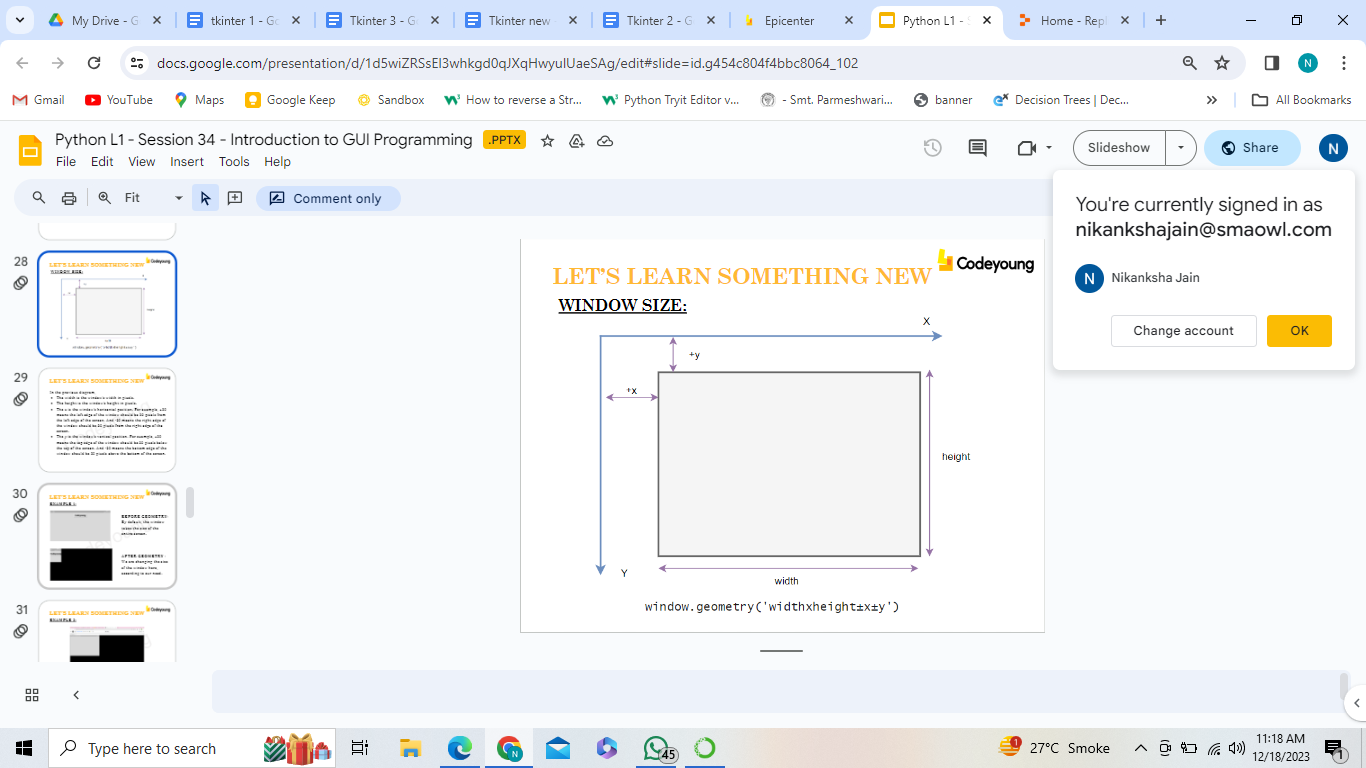
btn=Label(window,text="J=Hello, let's code")

btn.place(x=170,y=200)

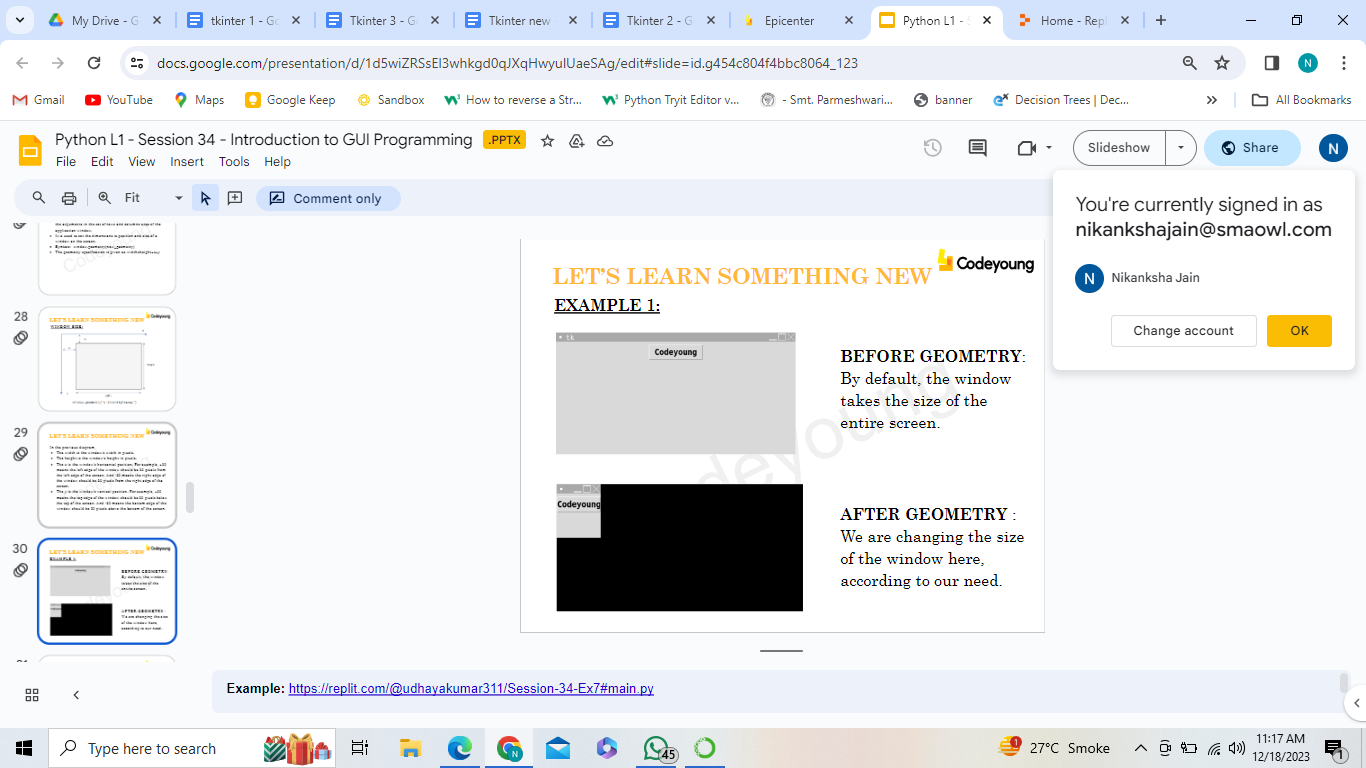
window.geometry('400x400')

window.mainloop()

**GEOMETRY METHOD**

* The geometry is one of the built-in methods and it will pass the arguments in the set of rows and columns area of the application window.
* It is used to set the dimensions ie.position and size of a window on the screen.
* **Syntax:** window.geometry(new\_geometry)
* The geometry specification is given as widthxheight±x±y
* 

In the previous diagram,

* The width is the window’s width in pixels.
* The height is the window’s height in pixels.
* The x is the window’s horizontal position. For example, +50 means the left edge of the window should be 50 pixels from the left edge of the screen. And -50 means the right edge of the window should be 50 pixels from the right edge of the screen.
* The y is the window’s vertical position. For example, +50 means the top edge of the window should be 50 pixels below the top of the screen. And -50 means the bottom edge of the window should be 50 pixels above the bottom of the screen
* 

**Extra examples:**

from tkinter import \*

import random

from tkinter import font

root = Tk()

root.geometry("500x500")

root.title("Dice")

root.config(bg="yellow")

def rollDice():

dice = 1 , 2 , 3 , 4 , 5 , 6

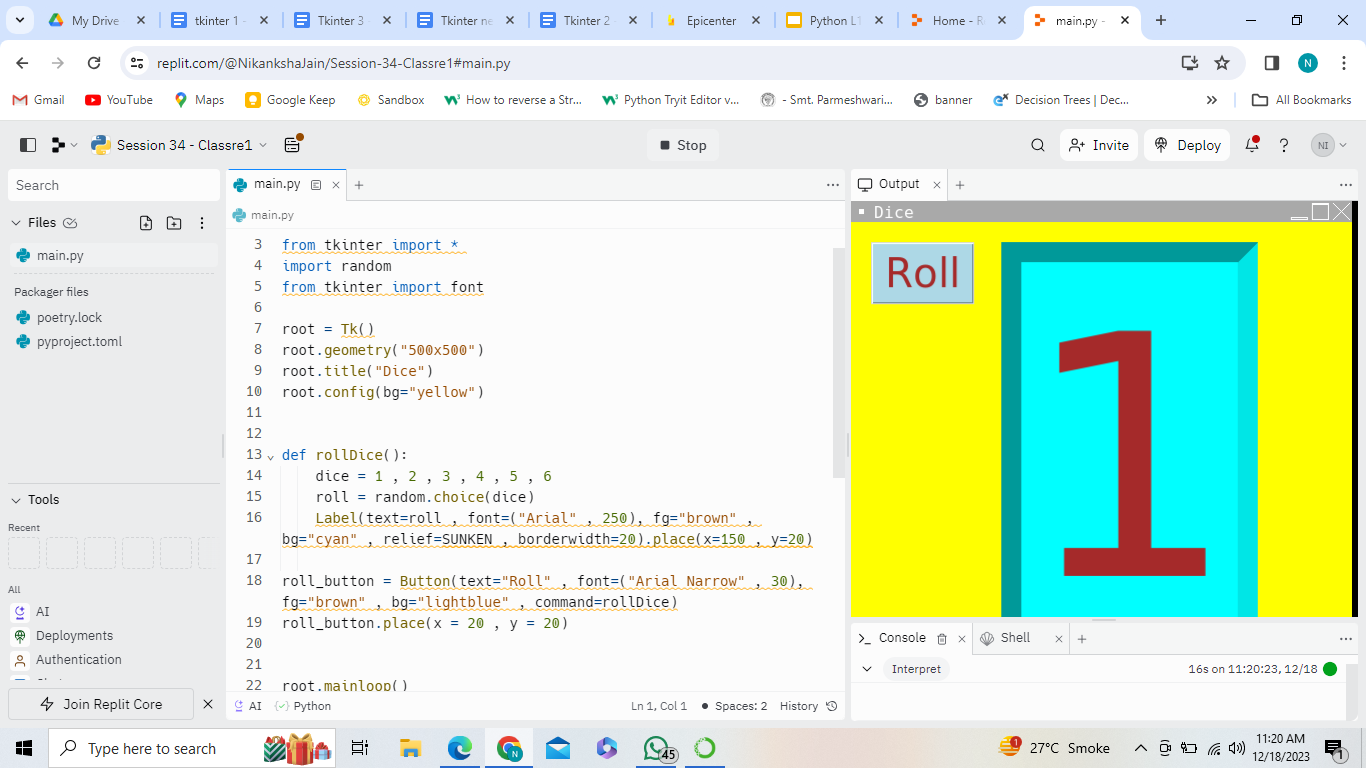
roll = random.choice(dice)

Label(text=roll , font=("Arial" , 250), fg="brown" , bg="cyan" , relief=SUNKEN , borderwidth=20).place(x=150 , y=20)

roll\_button = Button(text="Roll" , font=("Arial Narrow" , 30), fg="brown" , bg="lightblue" , command=rollDice)

roll\_button.place(x = 20 , y = 20)

root.mainloop()



<https://replit.com/@NikankshaJain/Session-34-Classre1#main.py>