PROJECT 1: SCREEN PET

1. Introduction:

Many people wish of having a fun, cute pet, now, what if you could get a virtual screen pet? It would help uplift your mood while working, just by looking at its face or stroking it, it might distract you from the stressful work.

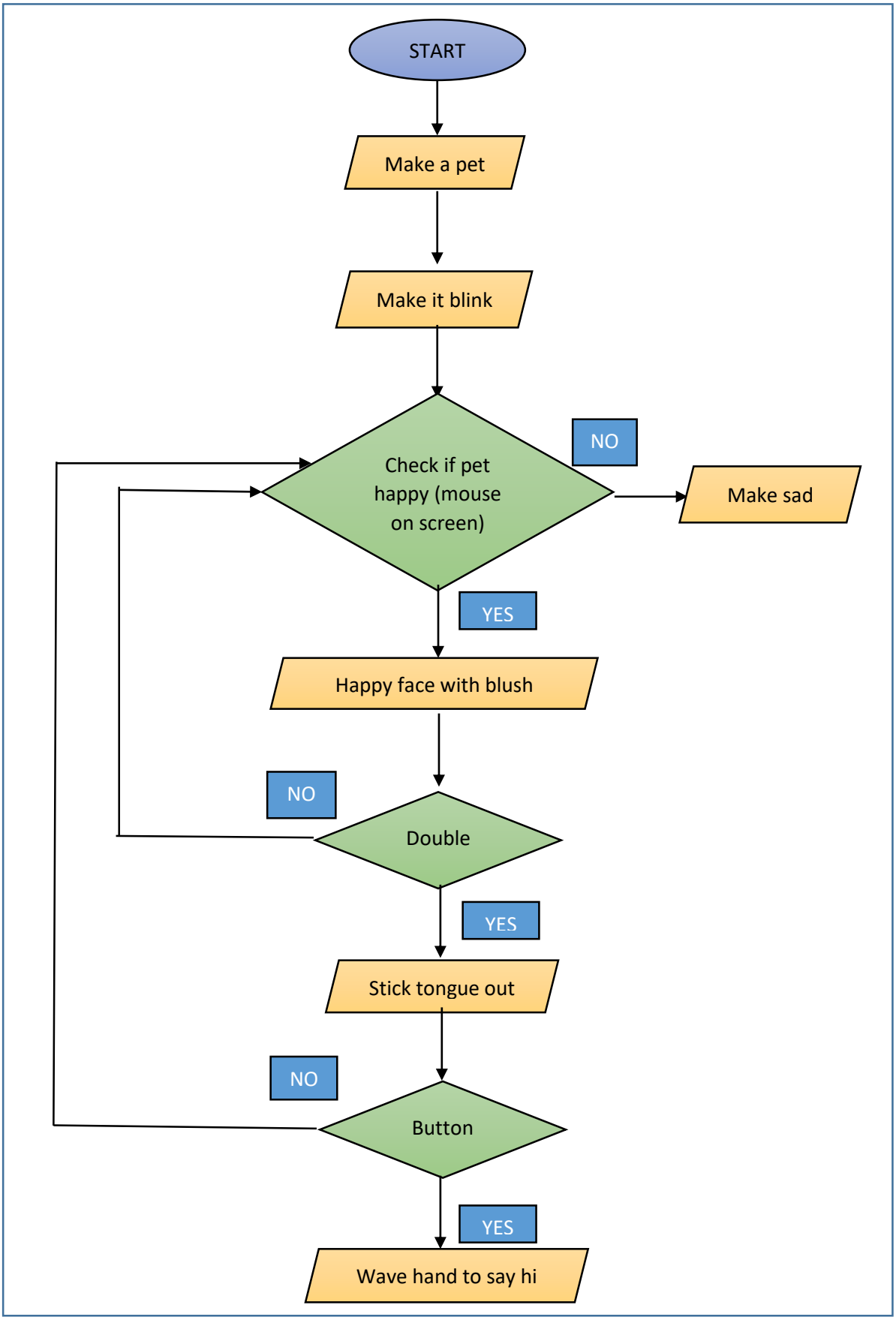
This application can provide you with a cheerful pet, which when ignored might get a little upset, so try not to make it unhappy. You can keep it on the corner of your screen to keep tabs on your pet. Kids would love trying to keep the pet happy and cheerful.

2. Technology Used:

For this application I have used **Tkinter** module available in python. This helps create a Graphic User Interface (GUI) for the application. This provides a canvas, buttons, labels, text, widgets and more. I created a screen pet aka panda pet, as shown below:

Fig.1: Initial display of the screen pet

3. Flowchart:


START
Make a pet
Make it blink
Check if pet happy (mouse on screen)
NO
Make sad
YES
Happy face with blush
NO
Double
YES
Stick tongue out
NO
Button
YES
Wave hand to say hi

4. Code for the project:

The GitHub repository link for the project is: https://github.com/mokshadashah/screen_pet

5. Explanation of the code:

a. Importing libraries:
Before running any program all the necessary libraries and modules should be imported into the program

```
from tkinter import *
```

b. Designing the canvas:
This shows the dimension of the window and background color of the canvas.

```
root = Tk()
# creating a canvas
c = Canvas(root, width=300, height=400)

# configuring its background
c.configure(bg='light sky blue', highlightthickness=0)
c.body_color = 'White'
```

c. Making the pet on the canvas:
Using various shapes of various sizes to make the screen pet
Some of the code is shown below:

```
#making the pet
left_ear = c.create_oval(40,71,120,153, fill = "black", outline = "black")
right_ear = c.create_oval(166,71,246,153, fill = "black", outline = "black")

body = c.create_polygon(92,259,114,234,207,212,196,263,173,307,148,312,123,308, fill = c.body_color

left_hand = c.create_polygon(59,254,84,224,114,235,85,266,77,268,66,268,60,264,59,258,59,252, fill =
right_hand = c.create_polygon(176,233,203,222,224,245,228,253,228,260,222,265,214,268,206,265,198,2

face = c.create_oval(40,88,244,242, fill = c.body_color, outline = "black", width = 2)

right_hand_up = c.create_polygon(190,228,218,217,233,199,254,195,257,214,244,234,201,259,200,244,19

tongue = c.create_oval(137,203,148,217, fill = "salmon", outline = "salmon", state = HIDDEN)
tongue_cover = c.create_oval(139,200,149,207, fill = c.body_color, outline = c.body_color)
```

d. Creating a function for making eyes blink:

```
# making a function to make the eyes blink
def toggle_eyes():
    current_state = c.itemcget(left_circle, 'state')
    new_state = HIDDEN if current_state == NORMAL else NORMAL
    c.itemconfigure(left_circle, state=new_state)
    c.itemconfigure(right_circle,state=new_state)
    c.itemconfigure(left_pupil, state = new_state)
    c.itemconfigure(right_pupil, state = new_state)
    c.itemconfigure(left_iris, state = new_state)
    c.itemconfigure(right_iris, state = new_state)
    c.itemconfigure(left_spark, state = new_state)
    c.itemconfigure(right_spark, state = new_state)

# calling the function to blink the eyes in another function
def blink():
    toggle_eyes()
    root.after(250, toggle_eyes)
    root.after(1000, blink)
```

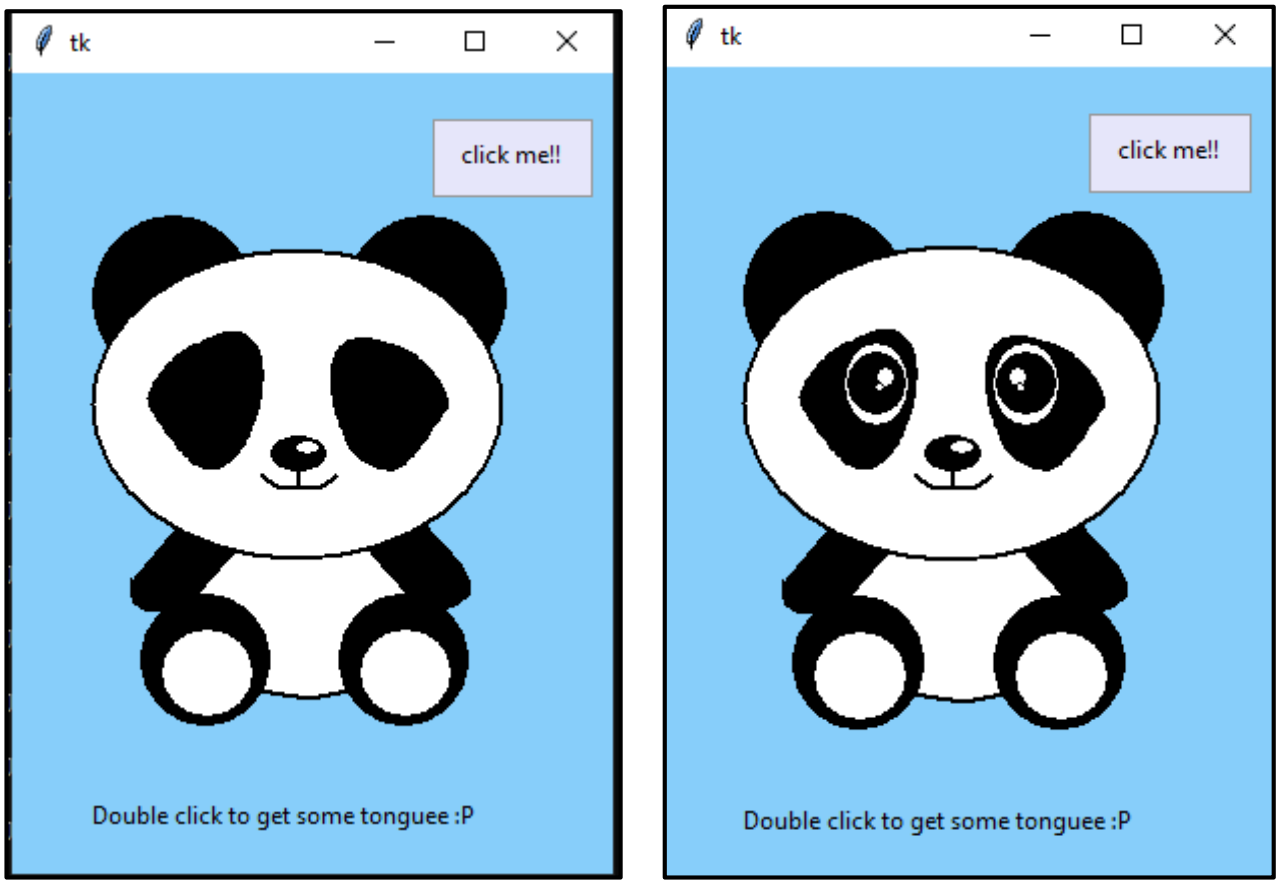


Fig.2: Blinking screen pet, shown in the above 2 images

e. Creating a function to make the pet happy and blush:

If the user moves the curser on the canvas, the pet will smile and blush.

```
# creating a function which will work on event which will
#make the pet blush if the mouse scrolls within the canvas
def show_happy(event):
    if (20 <= event.x <= 400) and (20 <= event.y <= 500):
        c.itemconfigure(left_cheek, state=NORMAL)
        c.itemconfigure(right_cheek, state=NORMAL)
        c.itemconfigure(happy_face, state = NORMAL)
        c.itemconfigure(sad_face, state = HIDDEN)
```

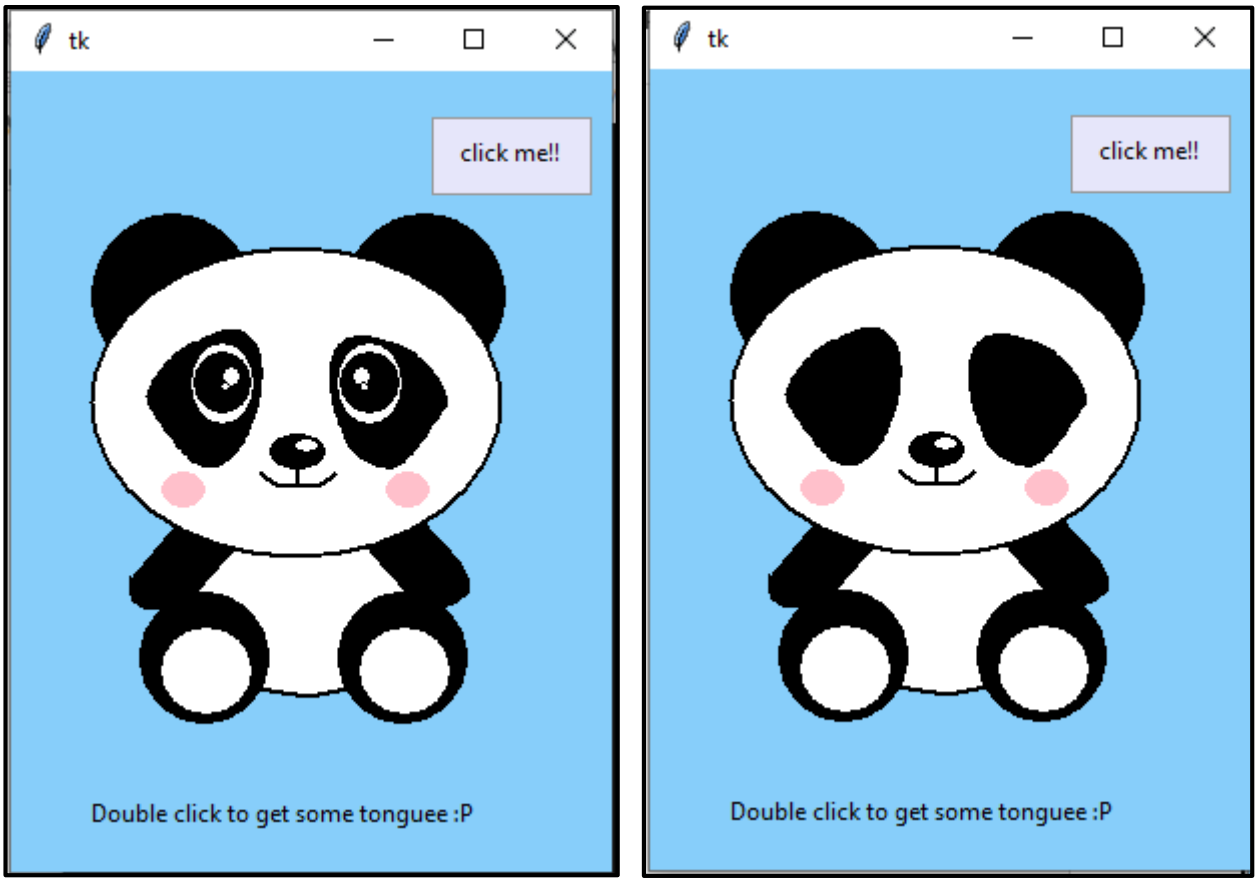


Fig.3: Happy Pet, with blush, when cursor is on the pet, with eyes open and closed, as it's always blinking

f. Creating a function to make the pet sad:

If the cursor isn't on the canvas, the pet will get sad.

```
# creating a function which will work on event
# if the mouse isnt on the canvas, it'll be sad
def hide_happy(event):
    c.itemconfigure(left_cheek, state=HIDDEN)
    c.itemconfigure(right_cheek, state=HIDDEN)
    c.itemconfigure(happy_face, state = HIDDEN)
    c.itemconfigure(tongue, state = HIDDEN)
    c.itemconfigure(sad_face, state = NORMAL)
```

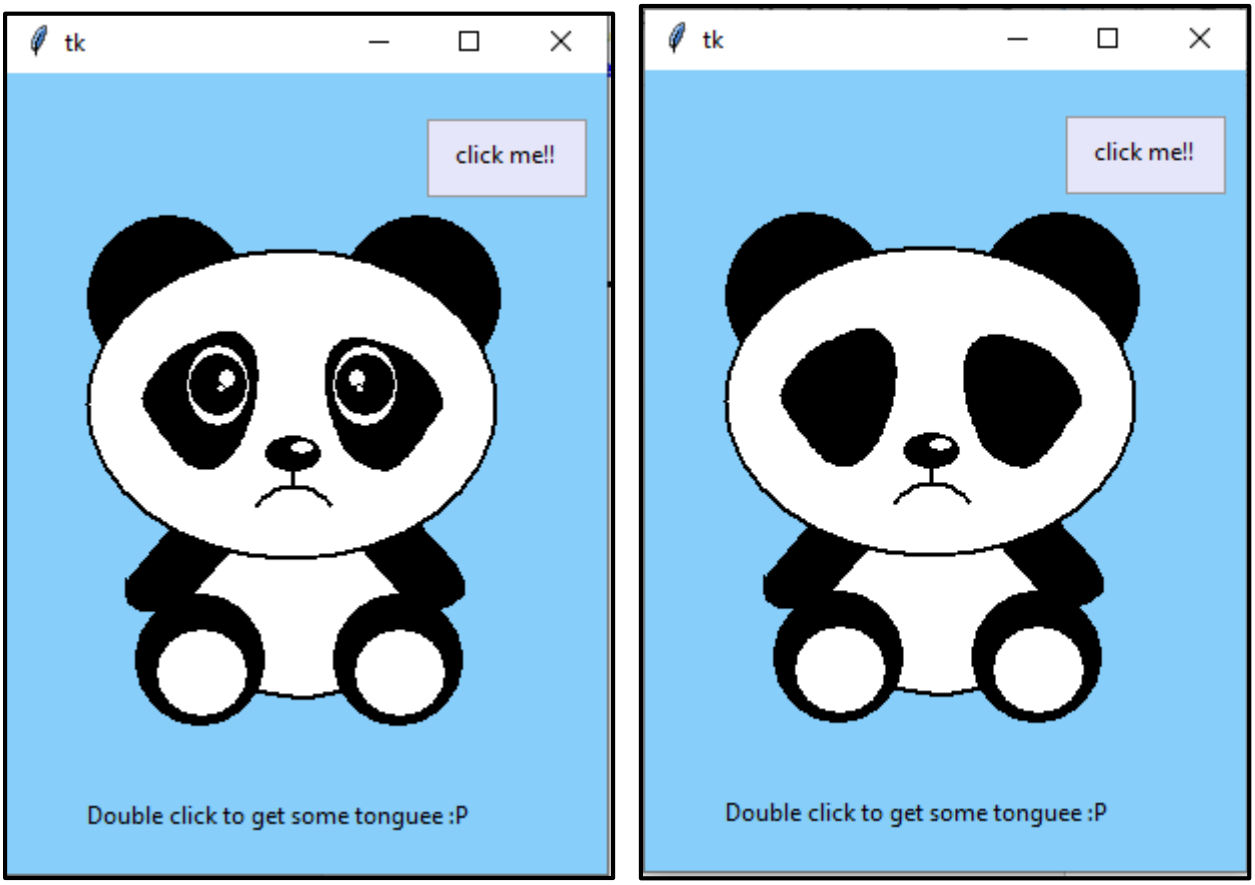


Fig.4: Sad pet, when cursor isn't on the pet, with eyes open and closed, as it's always blinking

g. Creating a function to make the show the tongue:

If the user, double clicks on the pet, the pet will show its tongue.

```
# making function to show the tongue
def toggle_tongue():
    c.itemconfigure(happy_face, state = NORMAL)
    c.itemconfigure(sad_face, state = HIDDEN)
    current_state = c.itemcget(tongue, 'state')
    new_state = NORMAL if current_state == HIDDEN else HIDDEN
    c.itemconfigure(tongue, state = new_state)

# creating a function which will work on event,
#if double clicked, it'll show tongue
def show_tongue(event):
        toggle_tongue()
        root.after(1000, toggle_tongue)
        c.itemconfigure(right_hand, state = NORMAL)
```

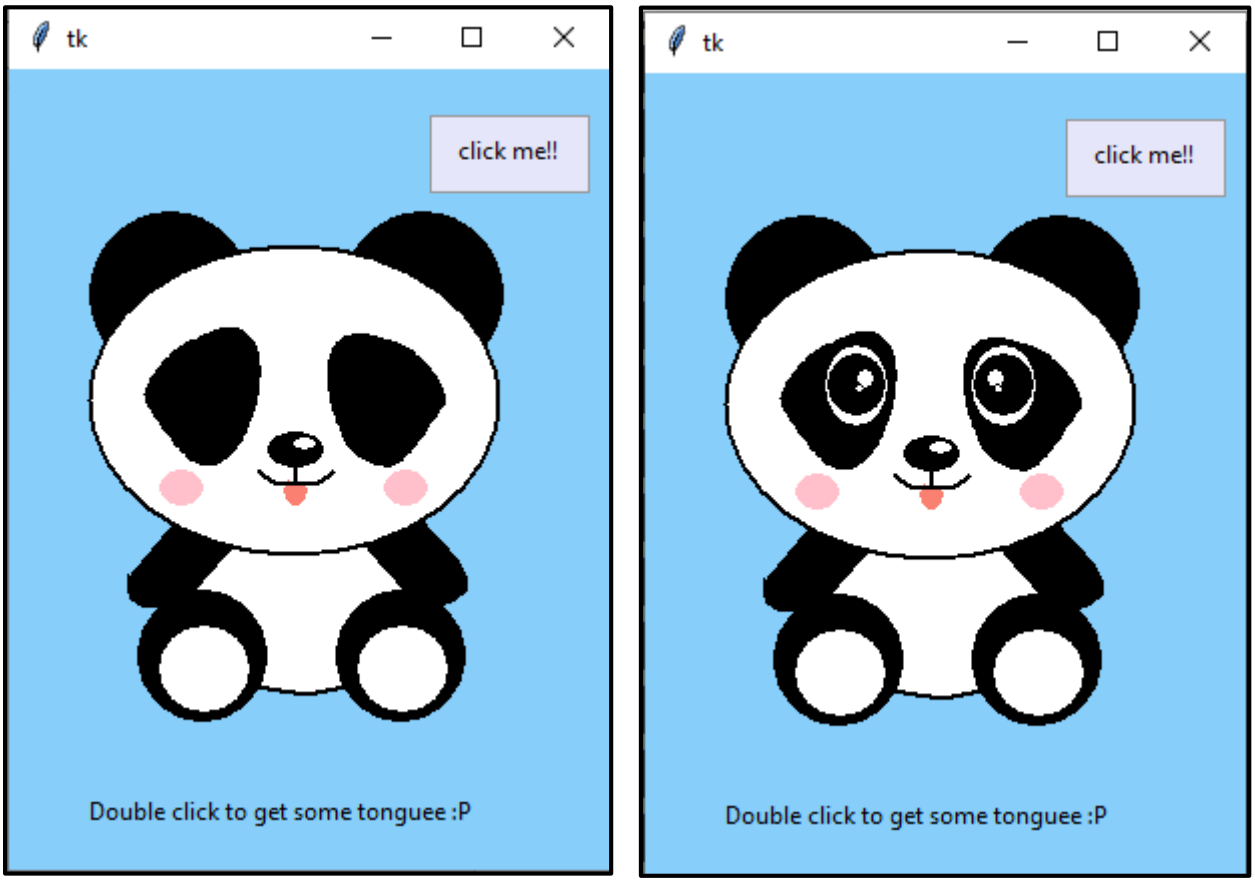


Fig.5: Tongue sticking out pet after double clicking, with eyes open and closed, as it's always blinking.

h. Creating a function to wave the hand:

```
# making function to show the the right hand wave
def toggle_hand():
    current_state = c.itemcget(right_hand_up, 'state')
    if current_state == HIDDEN:
            new_state = NORMAL
            c.itemconfigure(right_hand_up, state = new_state)
            c.itemconfigure(right_hand, state = HIDDEN)
    else:
        c.itemconfigure(right_hand_up, state = HIDDEN)
        c.itemconfigure(right_hand, state = NORMAL)


# creating a function which will make the hand wave on an event
def hand_wave(event):
    toggle_hand()
    root.after(1000, toggle_hand)
```

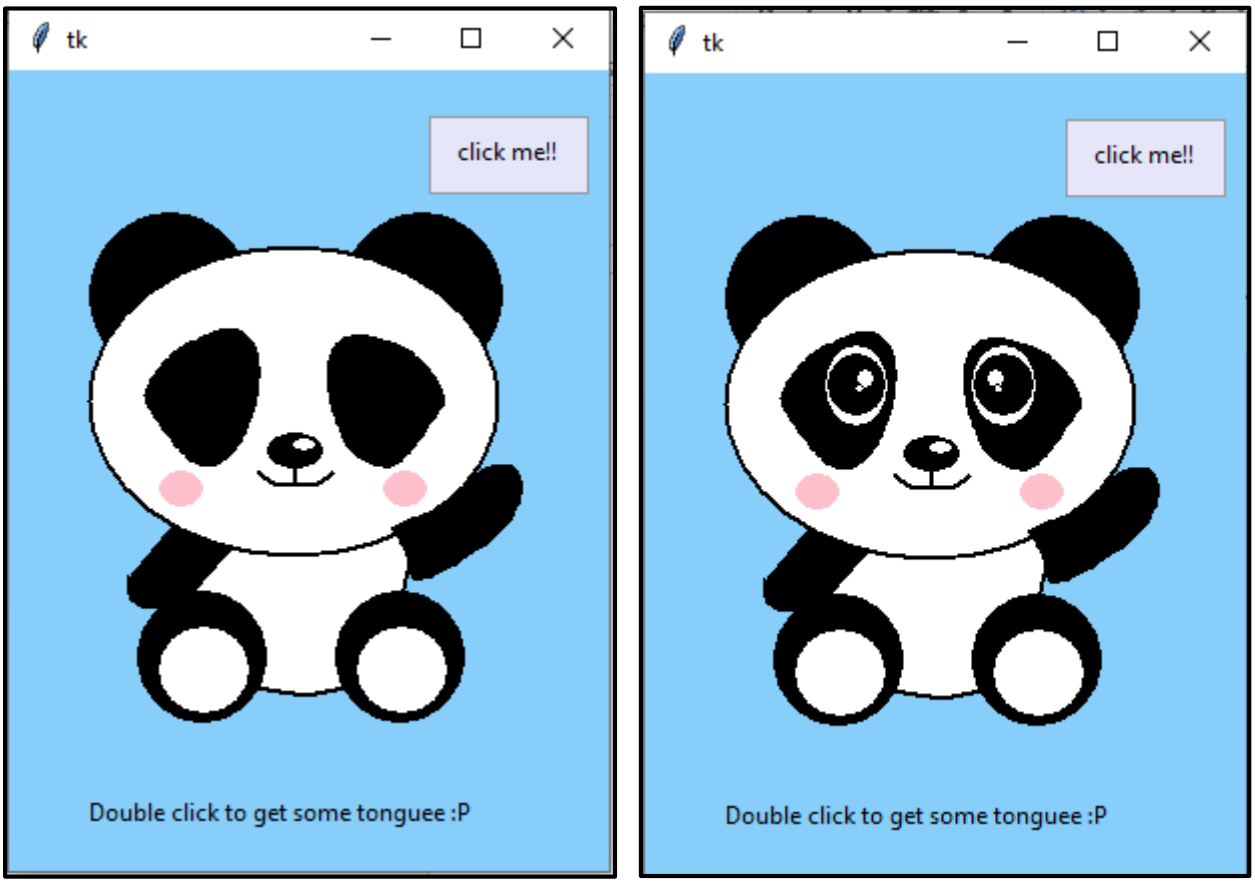


Fig.6: Upon clicking 'click me!!', the pet, waves it's hand with eyes open and closed, as it's always blinking.

i. Adding button, text and function to the button:

Adding hand wave function to the button and text created. And adding another text to make sure that the user double clicks to see some tongue.

```
# creating a button on the canvas
buttonBG = c.create_rectangle(210, 23, 289, 61, fill="lavender", outline="grey60")

# adding text inside the button
buttonTXT = c.create_text(249,40, text="click me!!")

#adding function (left click) on the button to make the hand wave
c.tag_bind(buttonBG, "<Button-1>", hand_wave)

# adding function (left click) on the text to make the hand wave
c.tag_bind(buttonTXT, "<Button-1>", hand_wave)

# adding text on canvas
tongueTXT = c.create_text(135,370, text="Double click to get some tonguee :P")
```

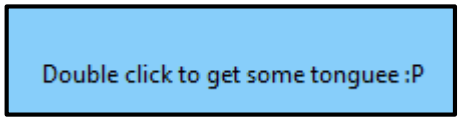


Fig.7: click me button **Fig.8: Text on the canvas**

j. Adding created function to the motions:

Making the pet, blink continuously, adding motion to show happy face with blush, make a sad face and to show tongue.

```
# continous blinking of the pet
root.after(20, blink)

# binding a motion which will show happy if there is motion on the canvas
c.bind('<Motion>', show_happy)

# if the mouse if left the canvas, it'll show sad face
c.bind('<Leave>', hide_happy)

# if double clicked, it'll show its tongue
c.bind('<Double-1>',show_tongue)

c.pack()
root.mainloop()
```