

# ভয়েস ক্যালকুলেটর কোড ব্যাখ্যা (বাংলা) - Step by Step ছোট ছোট ধাপে

## ১. লাইব্রেরি Import করা

```
import customtkinter as ctk
import re
import speech_recognition as sr
import threading
import time
import math
```

- **customtkinter (ctk):** Tkinter-এর modern version। Dark/Light mode, সুন্দর button ও entry তৈরি করতে সহায়ক।
- **re:** Regular expressions; input validation এবং special character handle করার জন্য।
- **speech\_recognition (sr):** ভয়েস ইনপুট নেয়ার জন্য।
- **threading:** UI hang না করে background এ voice recognition চালানোর জন্য।
- **time:** Animation delay ও live typing effect এর জন্য।
- **math:**  $\sqrt{\quad}$  (square root) এবং অন্যান্য math functions handle করতে।

যদি কোন লাইব্রেরি import না করা হয়, কোড run হবে না। প্রতিটি লাইব্রেরি স্পেসিফিক কাজের জন্য দরকার।

## ২. Safe Evaluation ফাংশন (safe\_eval)

```
def safe_eval(expr):
    try:
        expr = expr.replace("√", "math.sqrt")
        expr = expr.replace("^2", "**2")
        allowed = re.compile(r'^[0-9\.\+\-\*/\(\) mathsqrt]+\.$')
        if not allowed.match(expr):
            return "ERROR"
        result = eval(expr)
        if isinstance(result, float):
            result = round(result, 10)
        return result
    except:
        return "ERROR"
```

- **কারণ:** সরাসরি eval ব্যবহার করলে unsafe হতে পারে। এখানে শুধু safe characters allow করা হয়েছে।
- $\sqrt{\quad} \rightarrow \text{math.sqrt}$ ,  $^2 \rightarrow 2$ : Python execute করার জন্য।

- **ERROR handling:** ভুল input হলে user-friendly "ERROR" দেখাবে।

যদি allowed regex ঠিক না রাখা হতো, security risk হতে পারত।

### ৩. Button Click Handler (on\_button\_click)

```
def on_button_click(value):
    current_text = entry_field.get()

    if value == "C":
        entry_field.delete(0, ctk.END) # Clear সব input

    elif value == "DEL":
        if current_text == "ERROR":
            entry_field.delete(0, ctk.END)
        elif len(current_text) > 0:
            entry_field.delete(len(current_text)-1, ctk.END) # last character
delete

    elif value == "=":
        if current_text.strip() == "": return
        result = safe_eval(current_text) # expression calculate
        entry_field.delete(0, ctk.END)
        entry_field.insert(ctk.END, str(result))

    elif value == "x²":
        entry_field.insert(ctk.END, "²") # square operator add

    elif value == "√":
        entry_field.insert(ctk.END, "√") # square root operator add

    else:
        entry_field.insert(ctk.END, value) # normal number/operator add

    entry_field.xview_moveto(1) # Cursor সবসময় শেষে থাকবে
```

- প্রতিটি button আলাদা কাজ করে।
- Cursor সবসময় শেষে রাখলে user experience ভালো হয়।
- **অপশনাল:** যদি Cursor না শেষের দিকে রাখা হতো, user input confuse হতো।

## 8. Voice Button Animation

```
def animate_voice_button(stop_event):
    colors = ["#1f6aa5", "#2f82c5", "#4da3e0"]
    i = 0
    while not stop_event.is_set():
        voice_btn.configure(fg_color=colors[i % len(colors)])
        i += 1
        time.sleep(0.3)
    voice_btn.configure(fg_color="#2c506b") # Reset
```

- **কেন:** Listening অবস্থায় user দেখবে button animation।
- **stop\_event:** Thread safe animation বন্ধ করার জন্য।
- **অন্য color দিলে:** aesthetic impact পরিবর্তন হতে পারে।

## ৫. Voice Command Handling

```
def voice_command_thread():
    stop_event = threading.Event()
    animation_thread = threading.Thread(target=animate_voice_button,
    args=(stop_event,), daemon=True)
    animation_thread.start()

    voice_btn.configure(text="Listening...", font=("Consolas", 16, "bold"))

    recognizer = sr.Recognizer()
    try:
        with sr.Microphone() as source:
            audio_data = recognizer.listen(source, timeout=5,
            phrase_time_limit=8)
            command_text = recognizer.recognize_google(audio_data, language="bn-
            BD")

            replacements = {"যোগ":"+", "বিয়োগ":"-", "বিয়োগ":"-", "গুণ":"*",
            "ভাগ":"/",
                                "plus":"+", "minus":"-", "multiply":"*",
            "times":"*", "divide":"/",
                                "square":"^2", "sqrt":"√", "root":"√"}


            cmd = command_text.lower()
            for word, symbol in replacements.items():
                cmd = cmd.replace(word, symbol)
```

```

cmd = re.sub(r"^[^0-9\+\-\*\\/\(\)\^2\.]", "", cmd)

for char in cmd:
    entry_field.insert(ctk.END, char)
    entry_field.update()
    time.sleep(0.03)

except sr.UnknownValueError:
    entry_field.insert(ctk.END, " [Didn't catch that]")
except sr.RequestError:
    entry_field.insert(ctk.END, " [Speech error]")
except Exception as e:
    entry_field.insert(ctk.END, " [Error]")

stop_event.set()
voice_btn.configure(text=" Voice", font=("Consolas", 20, "bold"))

```

#### • Step by Step:

- Threading দিয়ে background এ voice recognition।
- Listening হলে button text change।
- Google API দিয়ে speech-to-text।
- Replace Bengali/English words to operators।
- Valid character filter।
- Entry field update digit-by-digit।
- Error handling।

যদি animation বা threading না থাকতো, UI hang হতো।

## ৬. GUI Setup

```

ctk.set_appearance_mode("dark")
ctk.set_default_color_theme("blue")
root = ctk.CTk()
root.title("Voice Calculator")

```

- Dark mode, theme setup।
- User-friendly appearance।

## Entry Field

```
entry_field = ctk.CTkEntry(root, width=280, height=50, font=("Consolas", 28, "bold"), justify="right", fg_color="#BBB", text_color="#000")
entry_field.grid(row=0, column=0, columnspan=4, pady=10, padx=5)
```

- Right aligned, readable font!
- Background color & height/width user-friendly!

## Buttons

- Top row: C, Voice, DEL
- Numbers & operators grid layout
- Lambda binding: `lambda t=text: on_button_click(t)` ensures correct function call.

---

## ৭. ভবিষ্যৎ উন্নতি

- বাংলা numbers recognition
- Calculation history দেখানো
- Keyboard input support
- Advanced math functions (trig, log, factorial)
- User-friendly error messages

প্রতিটি step ছোট ছোট ভাগে করলে debugging সহজ হয়।