

INTRODUCTION TO PROBLEM SOLVING AND PROGRAMMING

NOVEMBER 25, 2025

by:

NAME:SWOSTIK SHUBHAM

REG NO: 25MIP10045

BASIC IMAGE SLIDESHOW PLAYER WITH AUTOMATED TRANSITIONS

PROBLEM STATEMENT:

LIMITATIONS IN SIMPLE IMAGE VIEWING:

- Manual image navigation requires constant user intervention
- Basic image viewers lack automated slideshow capabilities
- No timed transitions between images
- Limited to viewing one image at a time
- Inefficient for presenting multiple images sequentially

PROJECT OBJECTIVE:







PRIMARY GOAL: TO DEVELOP A SIMPLE YET EFFECTIVE AUTOMATED IMAGE SLIDESHOW PLAYER THAT DISPLAYS IMAGES IN SEQUENCE WITH TIMED INTERVALS.

SPECIFIC OBJECTIVES:

- Create an automated image rotation system
- Implement timed transitions between images
- Provide basic user controls for slideshow management
- Support multiple image formats
- Ensure simple and intuitive user interface

SCOPE OF THE PROJECT:

IN-SCOPE FEATURES:

-  Automated image slideshow functionality
-  Fixed time interval transitions (3 seconds)
-  Basic user control (Play button)
-  Support for common image formats (PNG, JPG, WEBP)
-  Image resizing for consistent display
-  Continuous looping through images

OUT OF SCOPE FEATURES:

-  Animation effects and transitions

- ✗ Multiple control options (Pause, Previous, Next)
- ✗ Customizable timing intervals
- ✗ Image editing capabilities
- ✗ Multiple animation styles
- ✗ Advanced user interface elements

FUNCTIONAL REQUIREMENTS:

CORE FUNCTIONALITY:

- **Image Management**
 - Load multiple images from specified file paths
 - Support PNG, JPG, and WEBP formats
 - Automatic resizing to standard dimensions (1080x1080)
- **Slideshow Operations**
 - Automatic image progression every 3 seconds
 - Continuous cycling through image collection
 - Basic start functionality

USER INTERFACE:

- **Display Components**
 - Main image display area
 - Play button to initiate slideshow
 - Proper window sizing and title
- **Control Features**
 - Single-button control system
 - Visual feedback during operation

NON FUNCTIONAL REQUIREMENTS:

PERFORMANCE REQUIREMENTS:

- **Efficiency**
 - Smooth image loading and display
 - Consistent timing intervals
 - Minimal resource consumption
- **Reliability**
 - Stable operation during slideshow
 - Proper handling of image files
 - Continuous looping without interruptions

USABILITY REQUIREMENTS:

- **User Experience**
 - Simple and intuitive interface
 - Easy-to-understand controls

- Clear visual presentation
- **Accessibility**
 - Straightforward operation
 - Minimal learning curve
 - Consistent behavior

USABILITY REQUIREMENTS:

- **User Experience**
 - Simple and intuitive interface
 - Easy-to-understand controls
 - Clear visual presentation
- **Accessibility**
 - Straightforward operation
 - Minimal learning curve
 - Consistent behavior

TECHNICAL REQUIREMENTS:

- **Compatibility**
 - Python 3.x compatibility
 - Cross-platform functionality
 - Standard image format support
- **Maintainability**
 - Clean code structure
 - Easy to modify image paths
 - Simple to extend functionality

TECHNICAL IMPLEMENTATIONS:

TECHNOLOGIES USED:

- **Programming Language:** Python 3.x
- **GUI Framework:** Tkinter (Standard Python library)
- **Image Processing:** PIL/Pillow library
- **Cycle Management:** itertools.cycle

KEY COMPONENTS:

1. **Image Loading System**
 - Multiple format support
 - Automatic resizing
 - Path-based image loading
2. **Slideshow Engine**
 - Cyclic image iteration
 - Timed transitions using `root.after()`

- Continuous looping mechanism

3. User Interface

- Tkinter-based GUI
- Label widget for image display
- Button control for activation

PROJECT SIGNIFICANCE:

EDUCATIONAL VALUES:

- Demonstrates basic GUI application development
- Shows image processing with PIL library
- Illustrates event-driven programming concepts
- Teaches timed operations in Tkinter

PRACTICAL APPLICATIONS:

- **Personal Use:** Simple photo viewing
- **Educational:** Basic presentation tool
- **Retail:** Product image display
- **Exhibitions:** Automated image showcasing

CODE STRUCTURE OVERVIEW:

MAIN COMPONENTS OF THE CODE:

1. Image Preparation

```
image_paths = [...] # Image file paths
```

```
image_size = (1080, 1080) # Standard size
```

```
images = [Image.open(path).resize(image_size) for path in image_paths]
```

2. Slideshow Mechanism

```
slideshow = cycle(photo_images) # Infinite cycle
```

```
root.after(3000, update_image) # Timed transitions
```

3. Control System

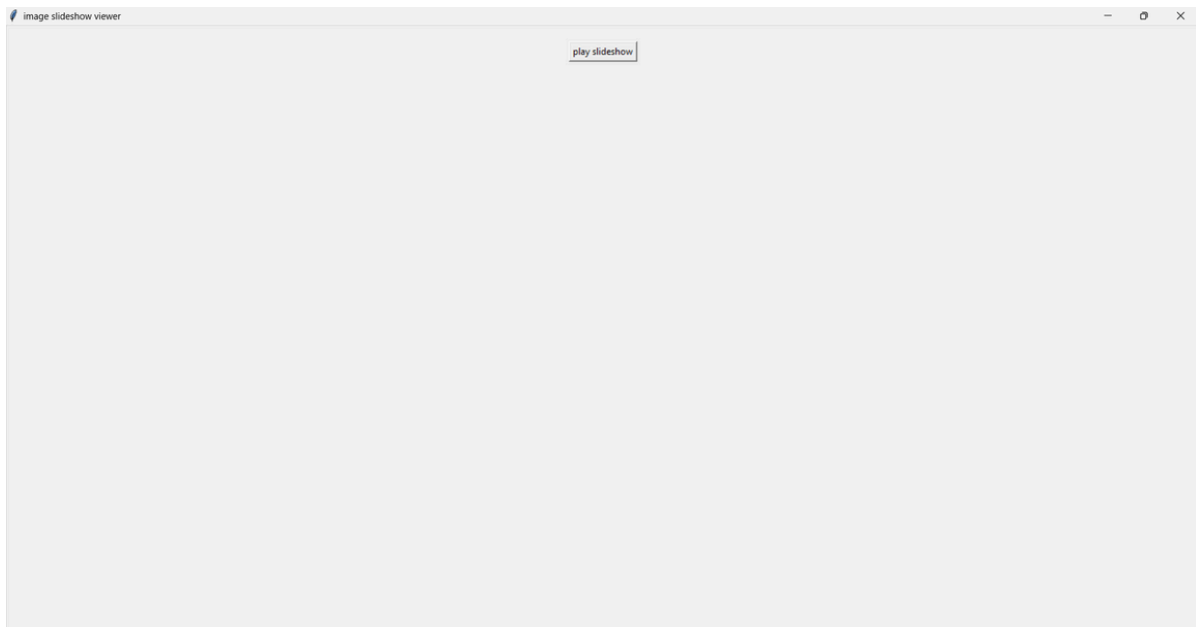
```
def start_slideshow(): # Initiation function
```

```
def update_image(): # Image update function
```

SCREENSHOTS:

```
python_project_2.py > ...
1  from itertools import cycle
2  from PIL import Image, ImageTk
3  import tkinter as tk
4
5
6  root= tk.Tk()
7  root.title("image slideshow viewer")
8  # list of image path
9  image_paths= [
10     r"C:\Users\HP\Pictures\Screenshots\Screenshot 2025-10-07 154659.png",
11     r"C:\Users\HP\Pictures\Screenshots\Screenshot 2025-10-24 204838.png",
12     r"C:\Users\HP\Pictures\209c67f1ba2d150991dbd846775e08ea.jpg",
13     r"C:\Users\HP\Pictures\captain-america-marvel-disney-movie-metal-07-11-2024-1731050780-hd-wallpaper.webp",
14     r"C:\Users\HP\Pictures\IMG-20251003-WA0021[1].jpg",
15     r"C:\Users\HP\Pictures\captain-america-marvel-disney-movie-metal-07-11-2024-1731050780-hd-wallpaper.webp",
16     r"C:\Users\HP\Pictures\Screenshots\Screenshot 2025-10-29 005122.png",
17     r"C:\Users\HP\Pictures\Screenshots\Screenshot 2025-11-02 115334.png",
18 ]
19
20
21 # now we will resize our image to 1080*1080
22 image_size= (1080,1080)
23 images= [Image.open(path).resize(image_size) for path in image_paths]
24 photo_images= [ImageTk.PhotoImage(image) for image in images]
25
26
27 label= tk.Label(root)
28 label.pack()
29 slideshow= cycle(photo_images)
30
31
32 def update_image():
33     photo_image= next(slideshow)
34     label.config(image = photo_image)
35     root.after(3000, update_image)
36
37
38 def start_slideshow():
39     update_image()
40
41
42 play_button= tk.Button(root, text= "play slideshow",command= start_slideshow)
43 play_button.pack()
44 root.mainloop()
```

OUTPUT WINDOW:



KEY FEATURES:

- **Automatic Cycling:** Uses itertools.cycle for endless looping
- **Timed Transitions:** 3-second intervals between images
- **Image Optimization:** Standardized 1080x1080 resolution
- **Simple Control:** Single-button operation

FUTURE ENHANCEMENT POSSIBILITIES:

POTENTIAL UPGRADES:

- Add pause/stop functionality
- Implement previous/next navigation
- Include customizable time intervals
- Add transition animations
- Incorporate image filtering effects
- Develop full-screen mode
- Add keyboard controls

CONCLUSION: THIS BASIC IMAGE SLIDESHOW PLAYER

SUCCESSFULLY DEMONSTRATES THE FUNDAMENTAL PRINCIPLES OF AUTOMATED IMAGE PRESENTATION. WHILE MAINTAINING SIMPLICITY, IT PROVIDES A FUNCTIONAL SOLUTION FOR SEQUENTIAL IMAGE DISPLAY WITH RELIABLE PERFORMANCE. THE PROJECT SERVES AS AN EXCELLENT FOUNDATION FOR UNDERSTANDING GUI DEVELOPMENT AND IMAGE PROCESSING IN PYTHON, WITH CLEAR POTENTIAL AND FUTURE EXPANSION AND ENHANCEMENT .